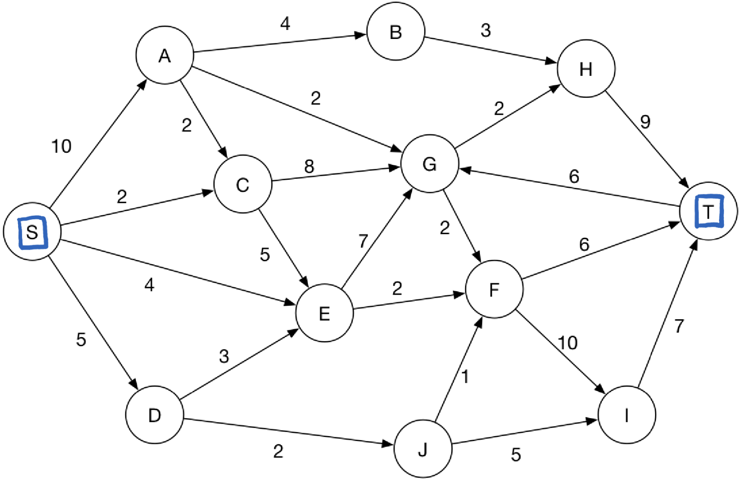
Some Games and Motivation

Source: Ahmadi, ORF 363 slides.

Let’s play some games (1)!

Let’s play some games (2)!

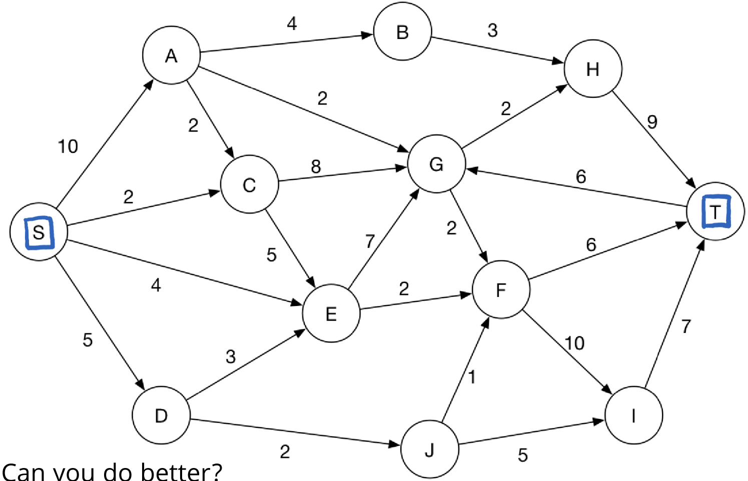
Modelling problem as a mathematical program



Rules:

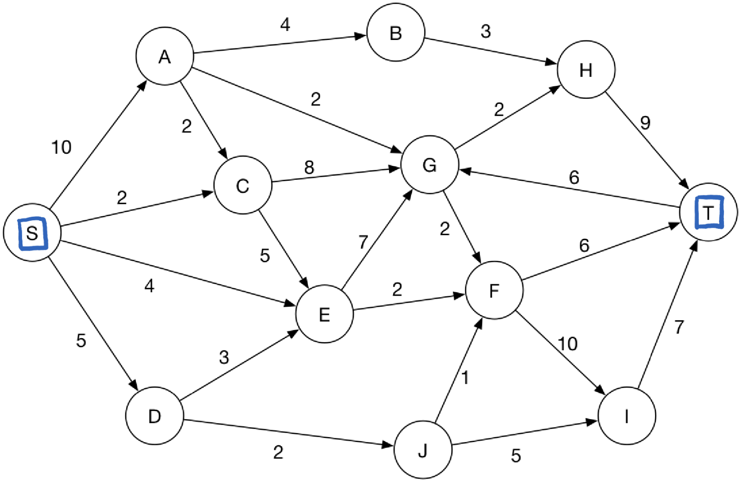
- Cannot exceed capacity of the edges
- Except S and T, flow in $=$ flow out
- **Goal:** Ship as much as possible from S to T

Let me give an example

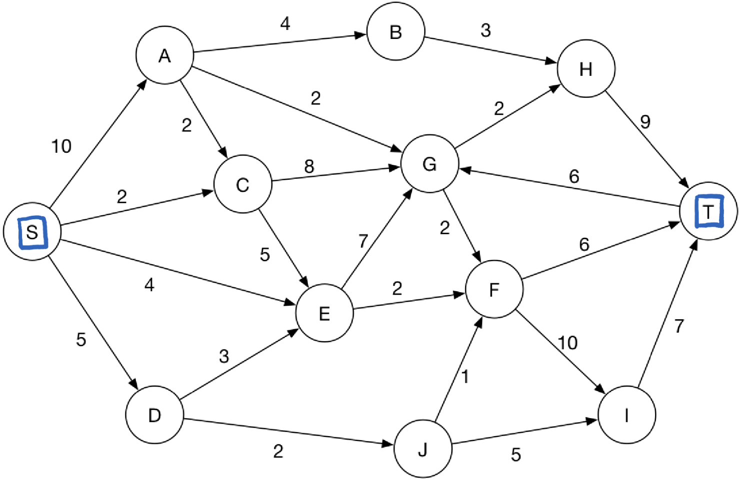


- Can you do better?
- Let’s try for 5 minutes

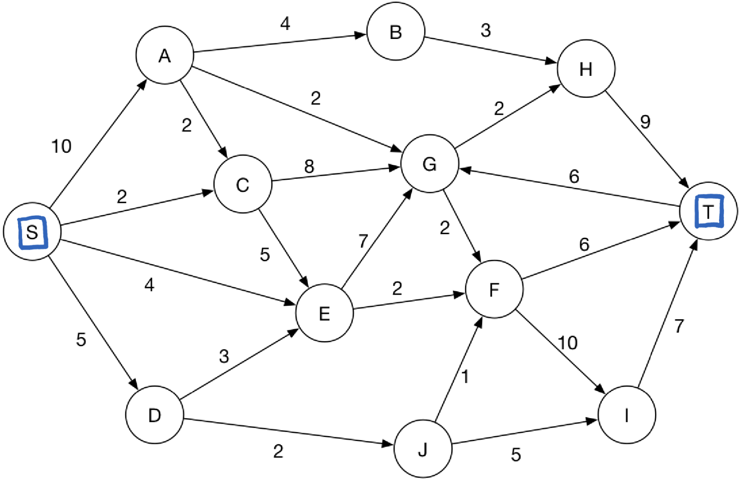
You tell me, I draw



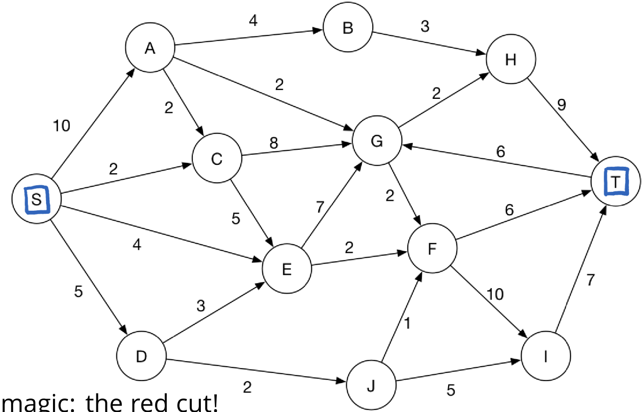
Can you do better?



Can you do even better?



11 is the best possible!

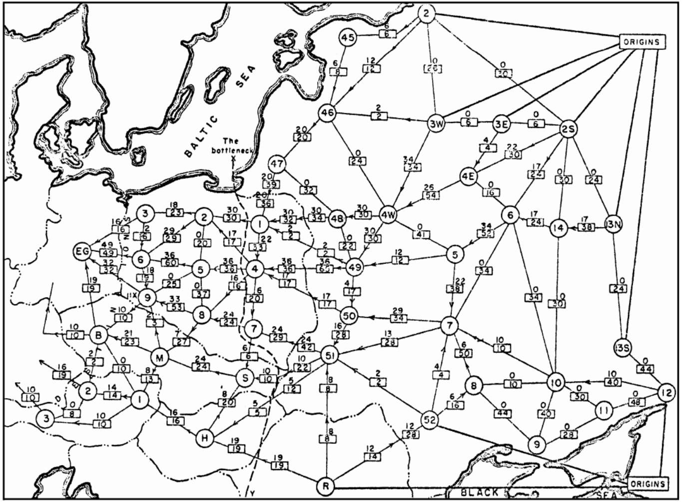


- The magic: the red cut!
- Any flow from S to T must cross red curve
- So it can have at most 11

Let’s try a more realistic graph

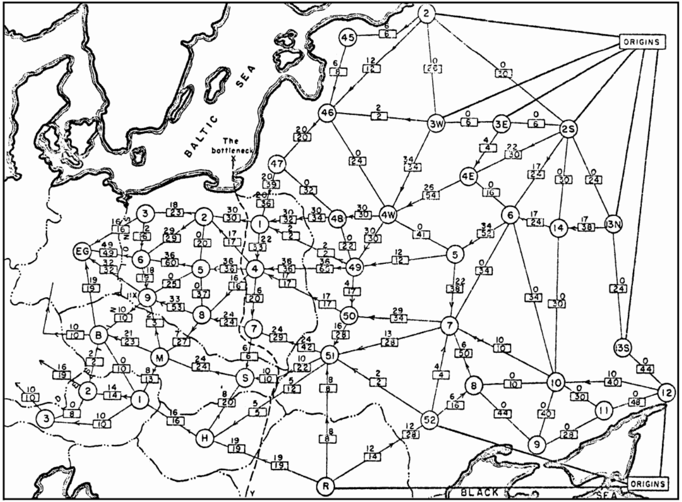
- Railway network of the Western Soviet Union going to Eastern Europe

- What’s the max capacity to transport cargo?



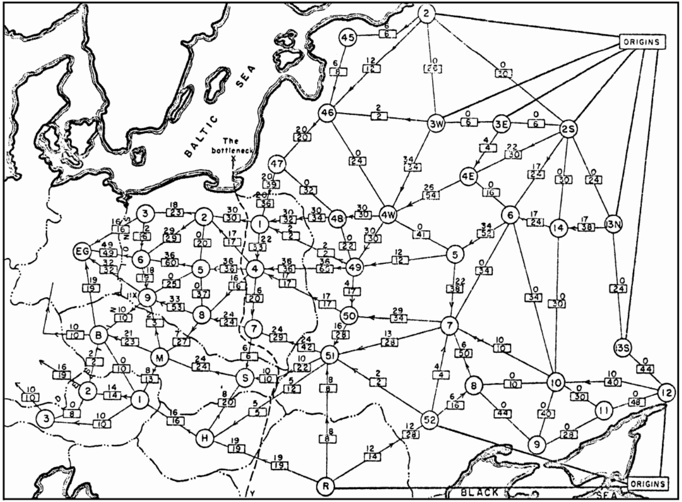
Let’s try a more realistic graph

- Railway network of the Western Soviet Union going to Eastern Europe
- What’s the max capacity to transport cargo?
- Look at the bottleneck!
- There are 44 vertices, 105 edges, and the max flow is 163K



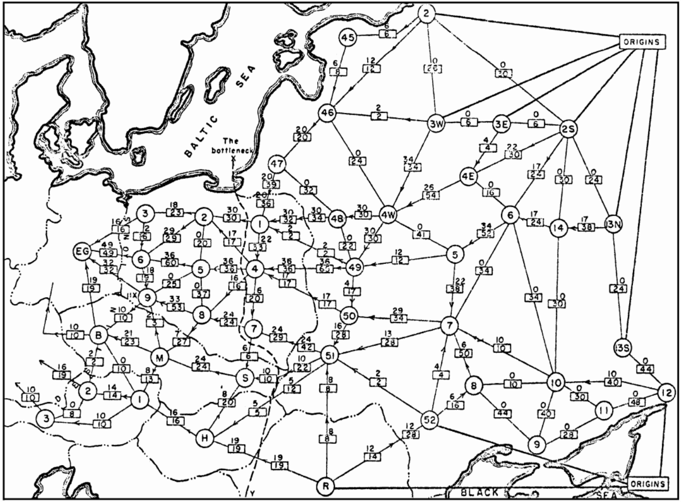
Let’s try a more realistic graph

- Is it possible to find the best solution?
- How long would it take (on my laptop)?
- How many lines of code?



Let’s try a more realistic graph

- Is it possible to find the best solution?
- How long would it take (on my laptop)?
- How many lines of code?
- Ford-Fulkerson algorithm (1955): always possible to find the best solution (for rational edge costs)



Let’s play some games (1)!

Let’s play some games (2)!

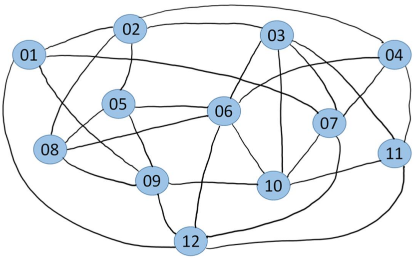
Modelling problem as a mathematical program

Two finals in one day? No thanks

- Department of Stats would like to schedule final exams for 12 courses

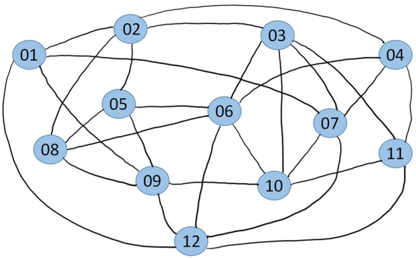
- Want to have as many exams as possible on the same day

- No student should have > 1 exam on that day



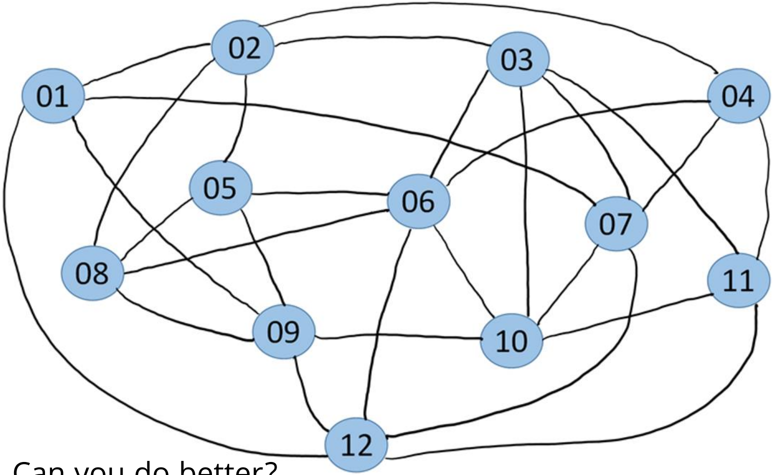
Two finals in one day? No thanks

- Department of Stats would like to schedule final exams for 12 courses



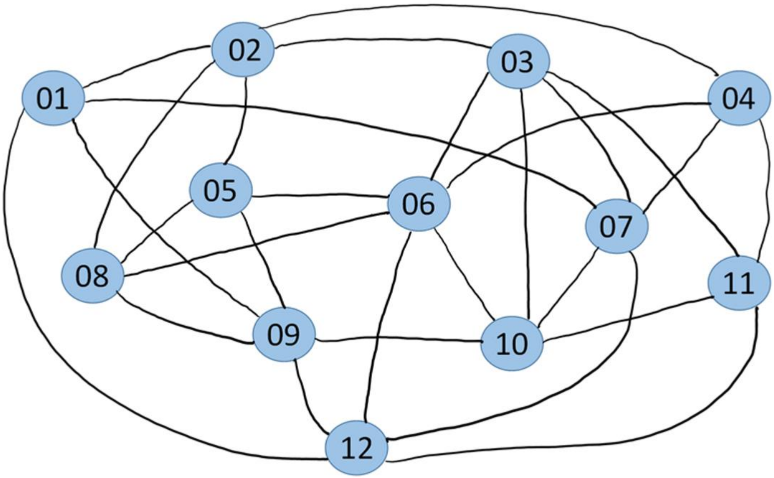
- Want to have as many exams as possible on the same day
- No student should have > 1 exam on that day
  - The nodes of this graph are 12 courses
  - edge between two nodes if there is at least one student taking both courses
  - Want to have as many exams as possible on a single day, what are we looking for?

Let me give you an example. Here is 3 concurrent exams:

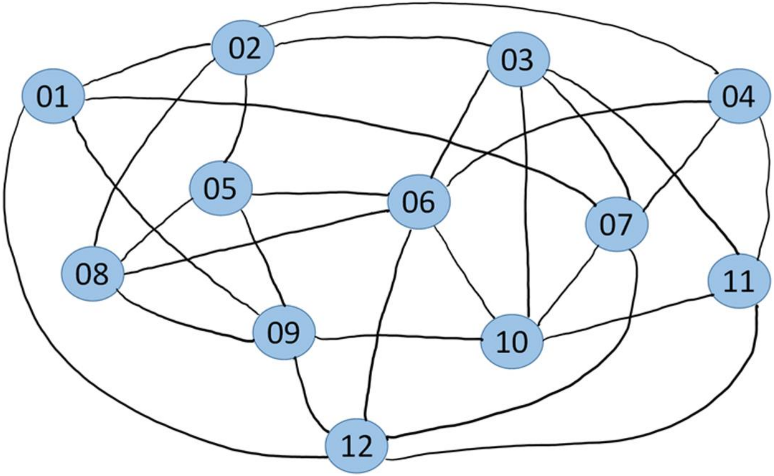


- Can you do better?
- Let’s try for 5 minutes

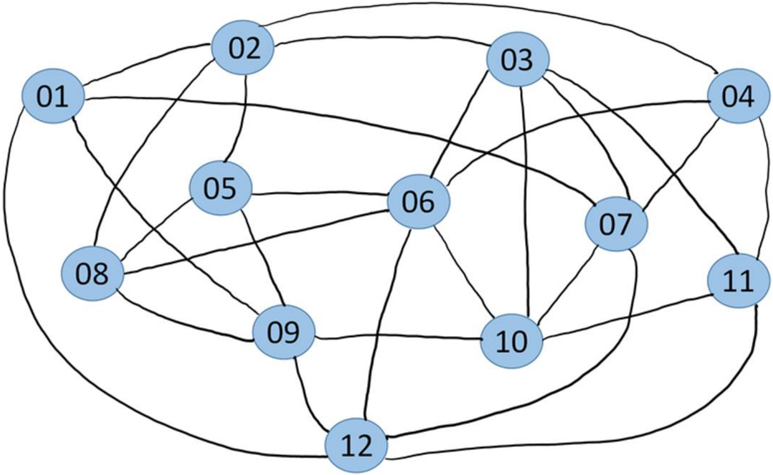
You tell me, I draw



You tell me, I draw



You tell me, I draw



5 is the best possible!

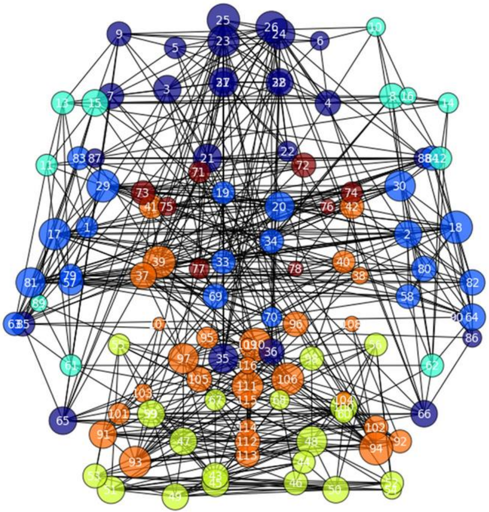
- Any magic? Unfortunately not :(
- No one has found a trick to this day

5 is the best possible!

- Any magic? Unfortunately not :(
- No one has found a trick to this day
- Want to find if 6 is possible?
  - Try all possible subsets of 6 nodes
  - There are 924 of them
  - Observe that none of them works

Let’s try another graph

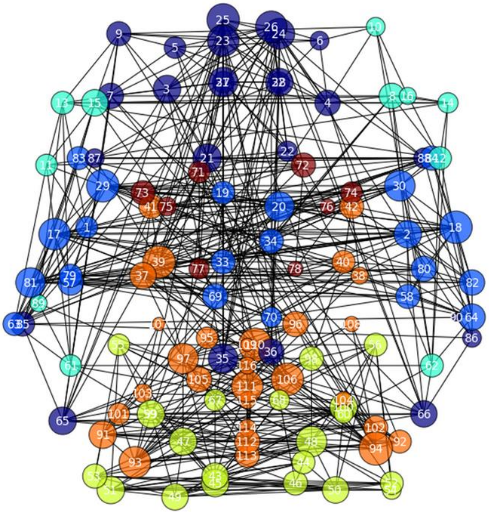
- Encouraged by the success of Stats Dep, now the Dean of Science wants to do the same for 115 SEAS courses



Let’s try another graph

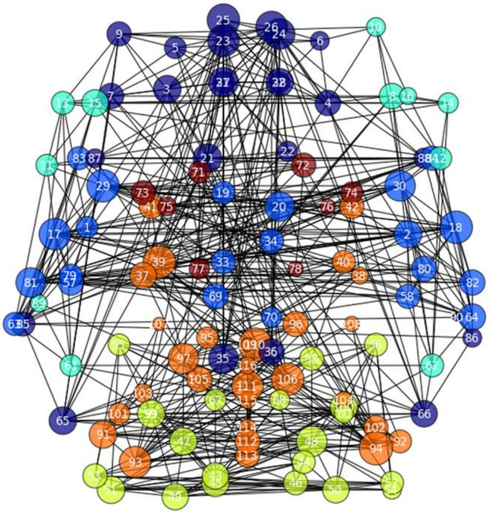
- Encouraged by the success of Stats Dep, now the Dean of Science wants to do the same for 115 SEAS courses

- How many final exams on the same day are possible? Can you do 17?



Let's try another graph

- Encouraged by the success of Stats Dep, now the Dean of Science wants to do the same for 115 SEAS courses
- How many final exams on the same day are possible? Can you do 17?
- Want to try out all possibilities for 17 exams?



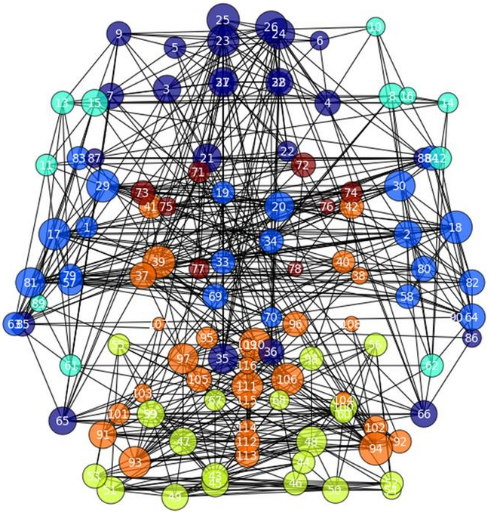
Let’s try another graph

- Encouraged by the success of Stats Dep, now the Dean of Science wants to do the same for 115 SEAS courses

- How many final exams on the same day are possible? Can you do 17?

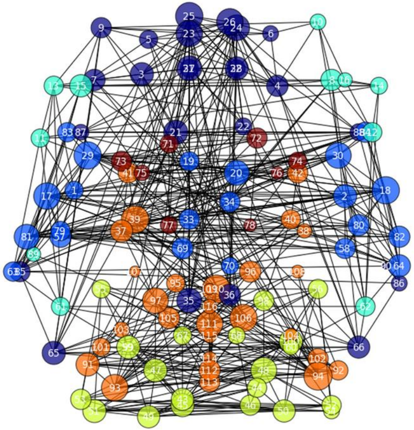
- Want to try out all possibilities for 17 exams?

- There are over 80000000000000000000 of them!



But there's some good news!

- Even though finding the best solution is not always possible, techniques from optimization (and in particular from the area of convex optimization) often allow us to find high-quality solutions with performance guarantees.



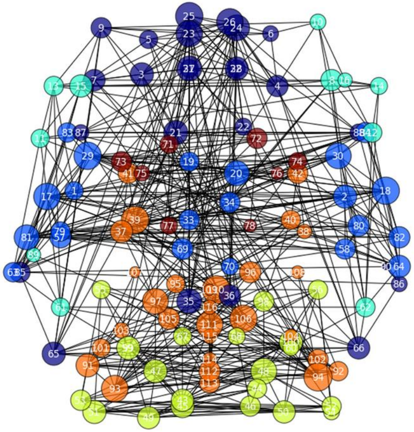
But there’s some good news!

- Even though finding the best solution is not always possible, techniques from optimization (and in particular from the area of convex optimization) often allow us to find high-quality solutions with performance guarantees.

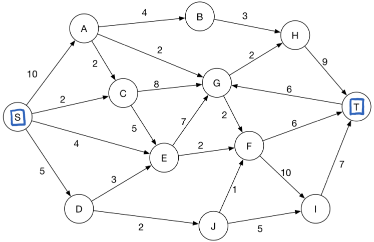
- For example, an optimization algorithm may quickly find 16 concurrent exams for you.

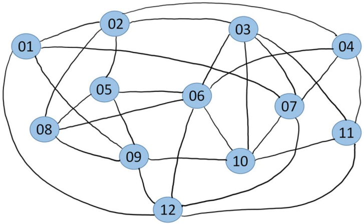
- Another optimization algorithm (or sometimes the same one) tells you that 19 is impossible.

- This is very useful information! You know you got 16, and no one can do better than 19.



Which problem was harder for you?





- Not always obvious. A lot of research in optimization goes into distinguishing the “tractable” problems from the “intractable” ones.

- The first problem we can solve efficiently (in “polynomial time”).

- The second problem: no one knows. If you do, you literally get $1M!

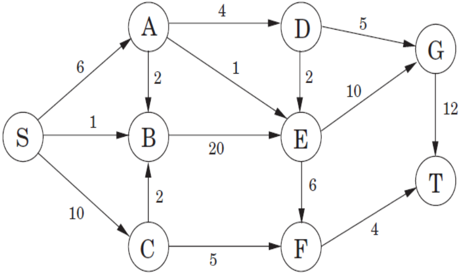
Let’s play some games (1)!

Let’s play some games (2)!

Modelling problem as a mathematical program

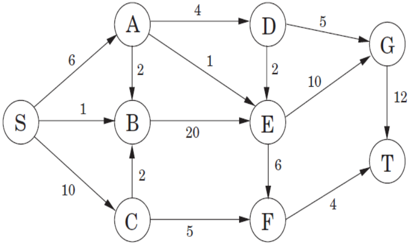
Let's revisit our first game

- What were the **variables**?
- What was your **objective function**?
- What were your **constraint**?



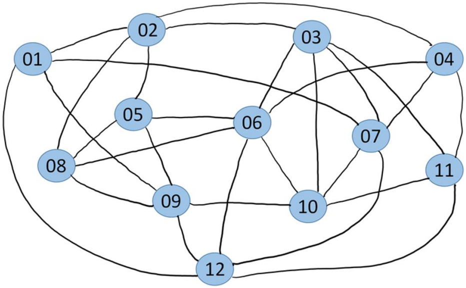
Rules:

- Cannot exceed capacity of the edges
- Except S and T, flow in $=$ flow out
- **Goal:** Ship as much as possible from S to T


S
A
B
C
D
E
F
G
T
6
1
10
4
2
1
20
2
5
2
10
6
5
4
12

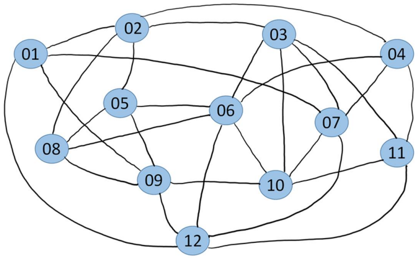
Let’s revisit our second game

- What were the **variables**?
- What was your **objective function**?
- What were your **constraint**?



Rules:

- **Goal:** Find maximum subset of nodes
- No connecting edge in the subset


02
03
01
04
05
06
07
08
11
09
10
12

Why one hard and one easy? How can you tell?

Caution: just because we can write something as a mathematical program, it doesn't mean we can solve it.

Course objectives

- Ability to view your own field through the lens of optimization and computation
- Ability to recognize hard and easy optimization problems
- Ability to use optimization software