Graph Modeling: Network Flows to Inform Course Selection

Zakk Heile (Duke 2027) and Isaac Yang (DKU 2026)

December 11, 2024

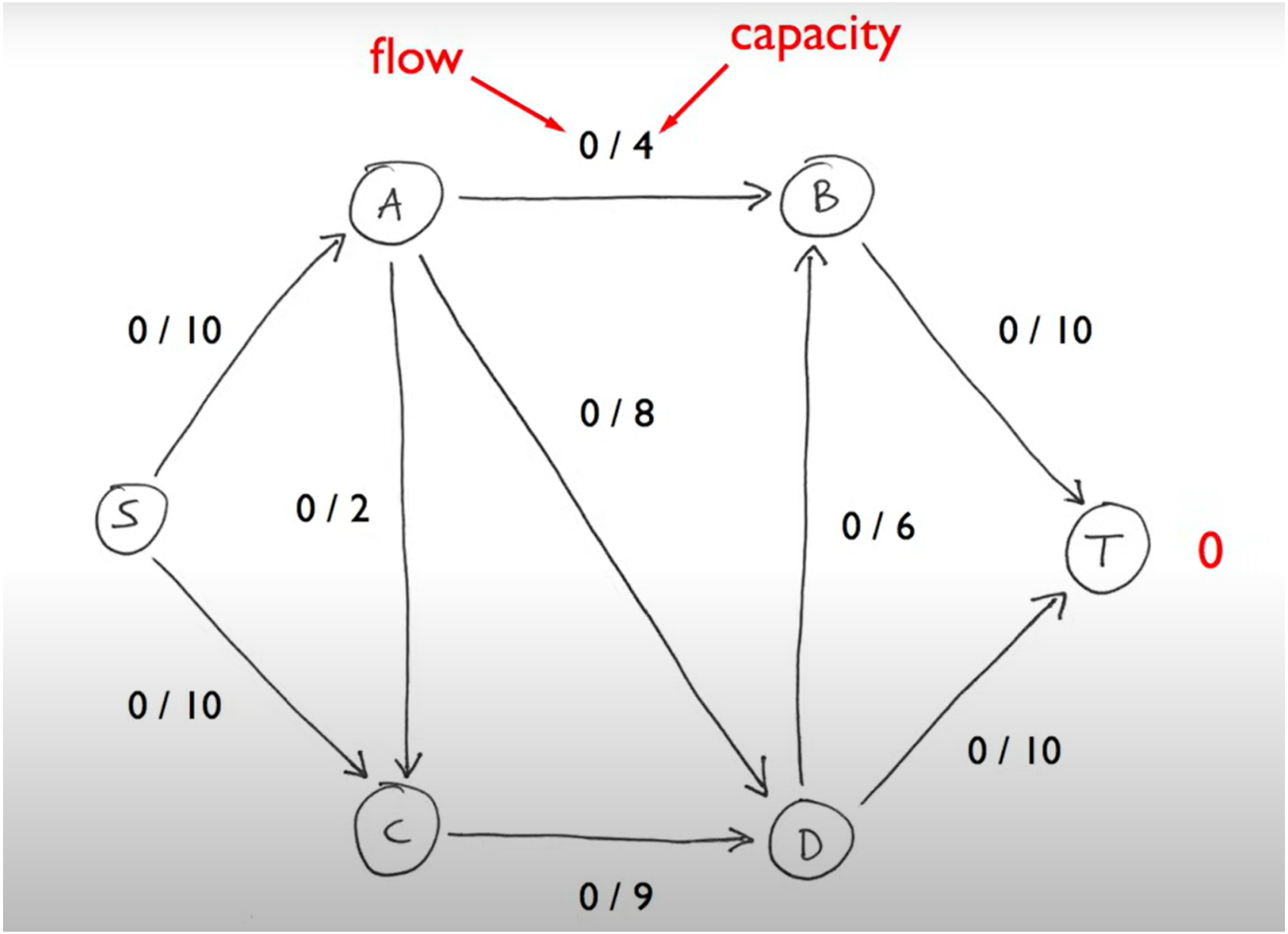
Overview of Problems

- Check if a selection of classes satisfies distribution requirements.
- Find the **minimum number** of classes to satisfy requirements.
- Optimize class selection to minimize time commitment or maximize enjoyment while satisfying requirements.
- Given a set of already-taken classes, determine optimal ways to satisfy constraints.

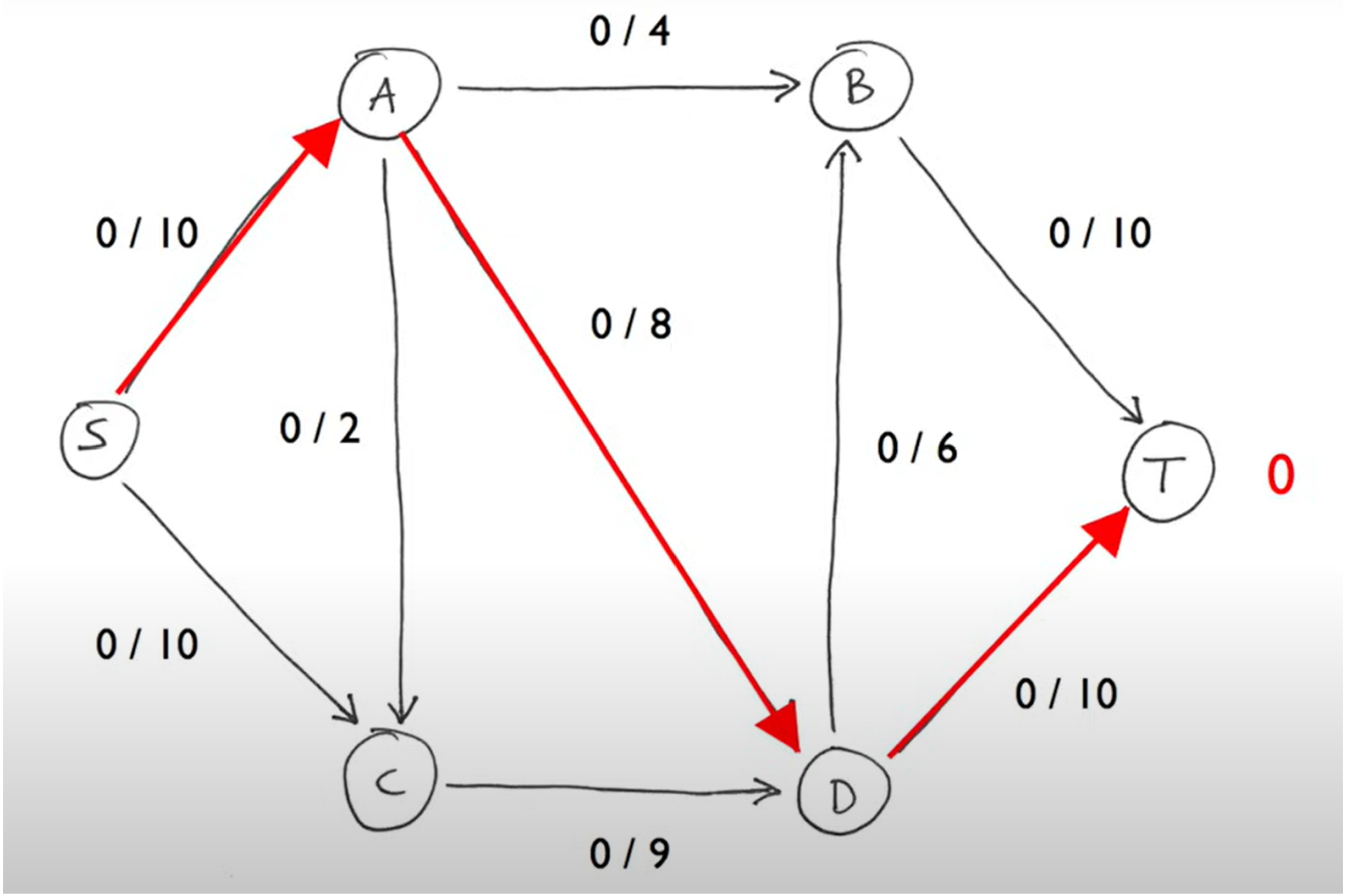
Z

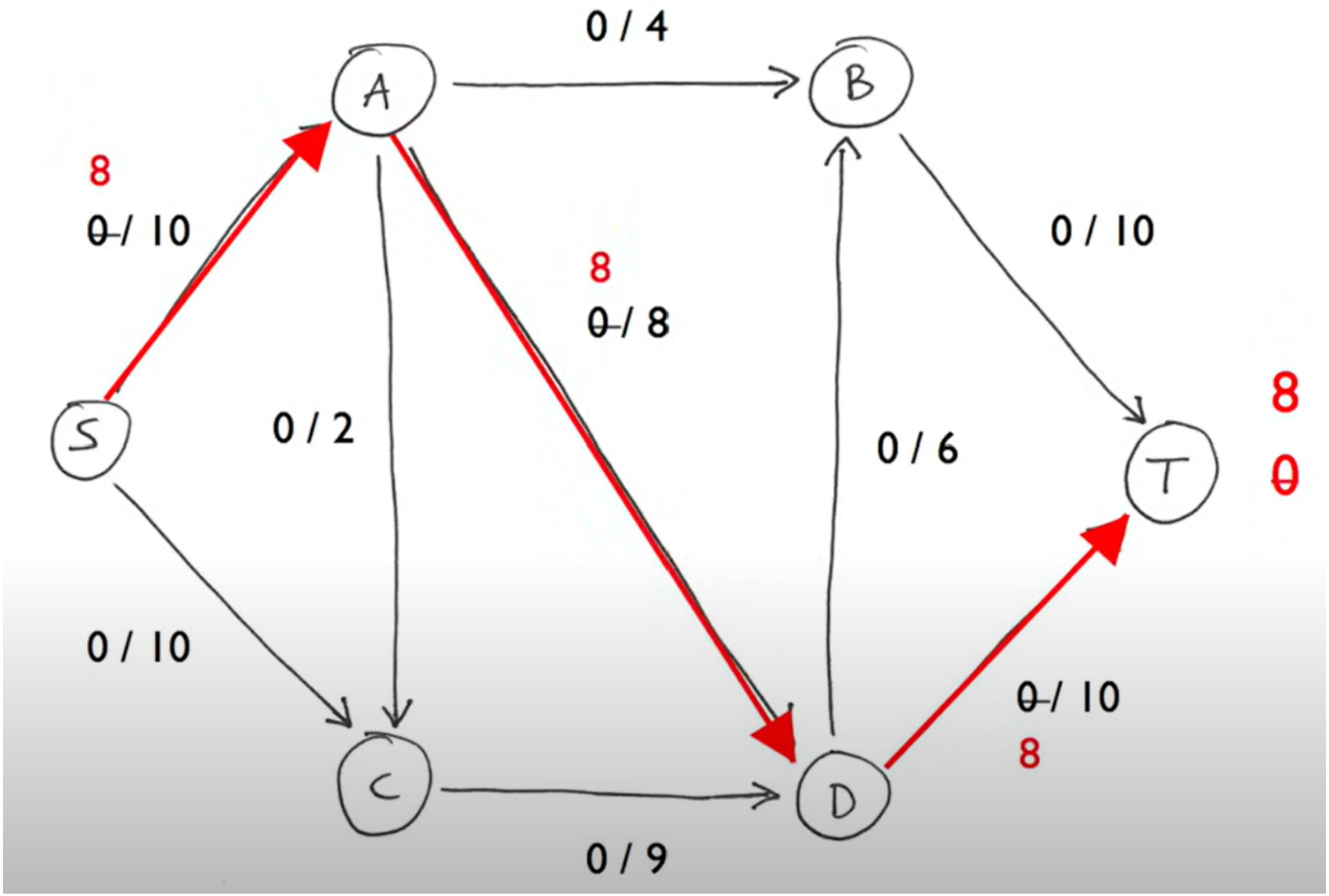
Modeling Course Selection

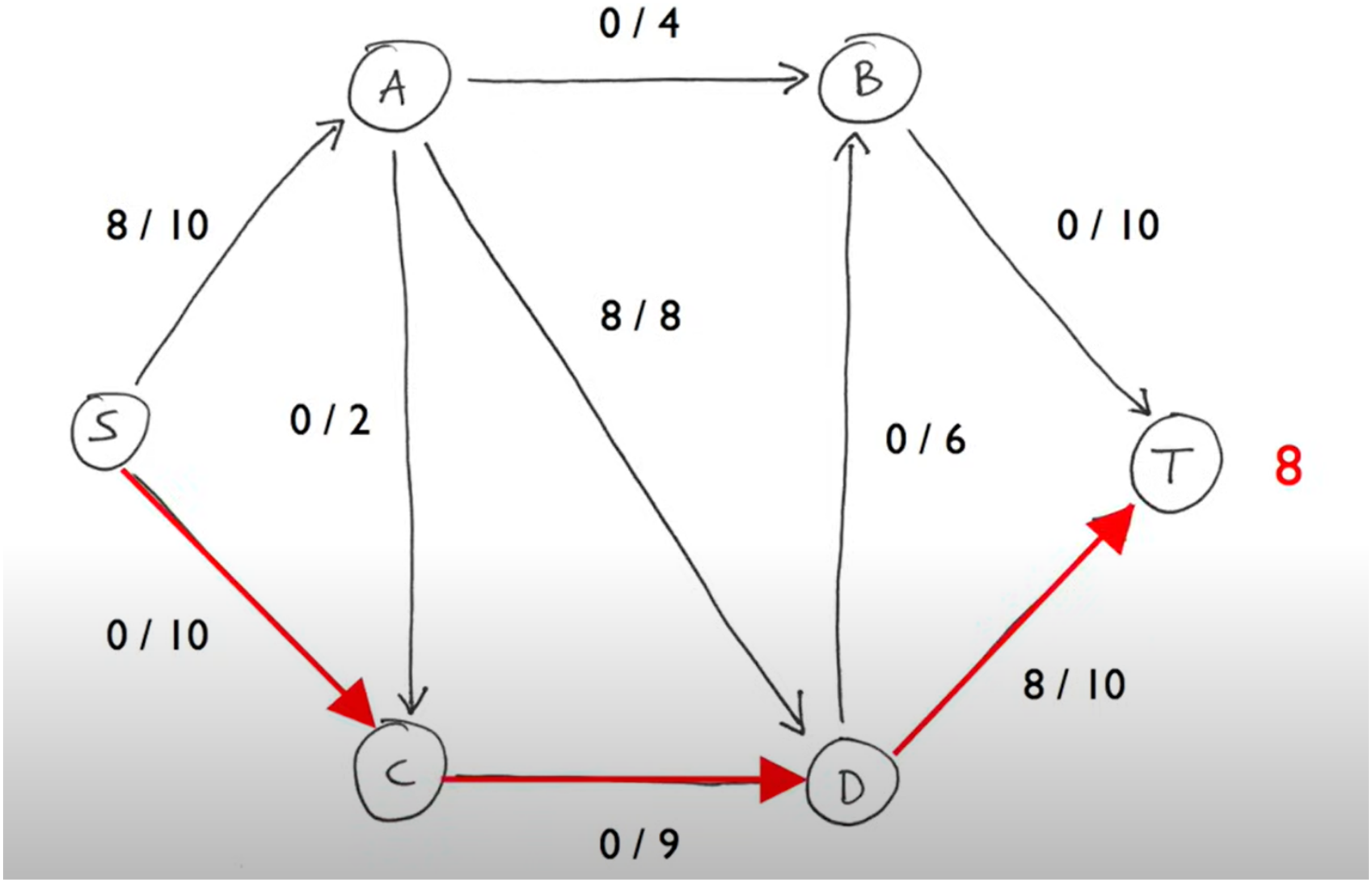
We model the Course Selection problem as a Max Flow problem. Z

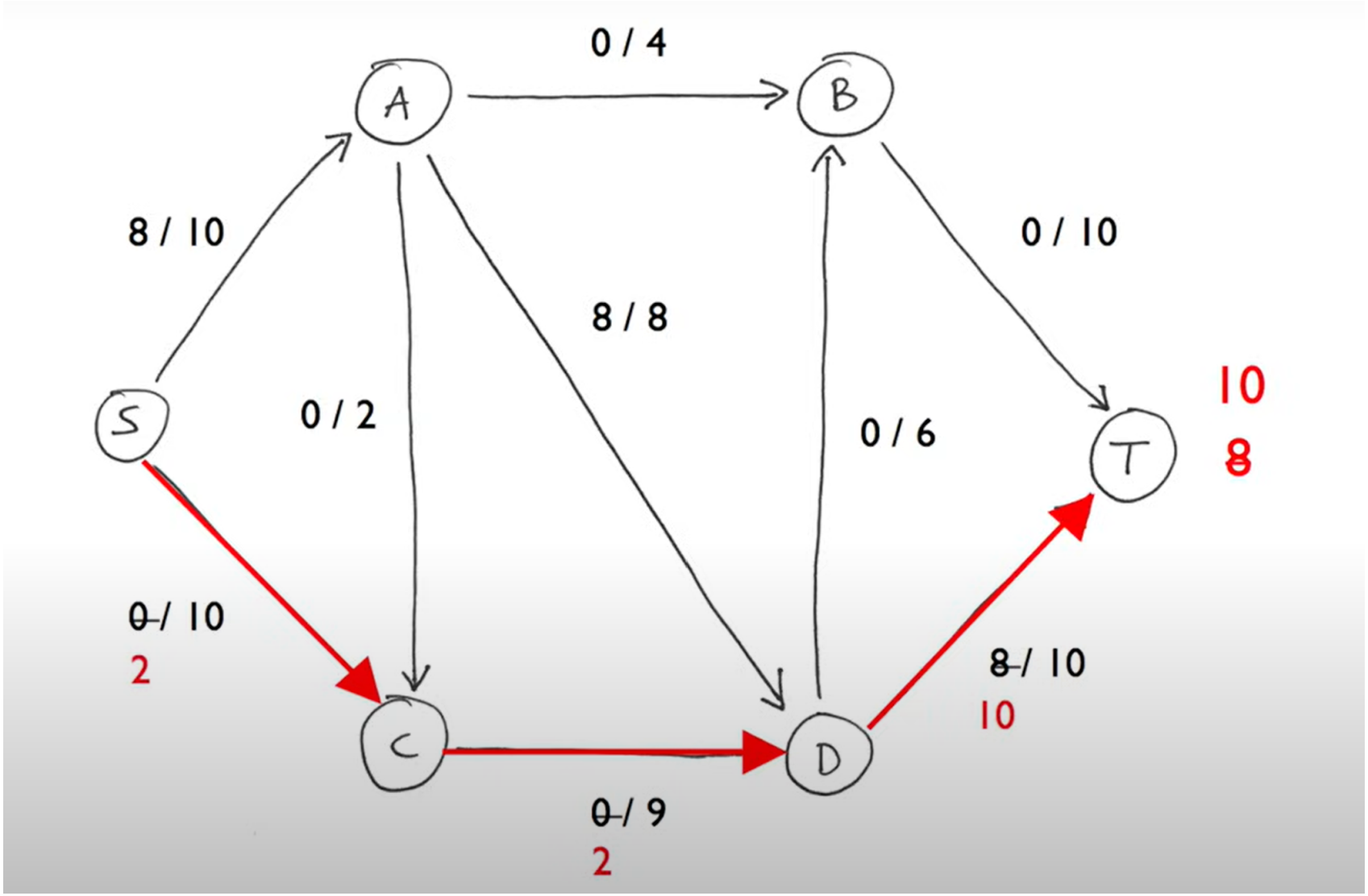


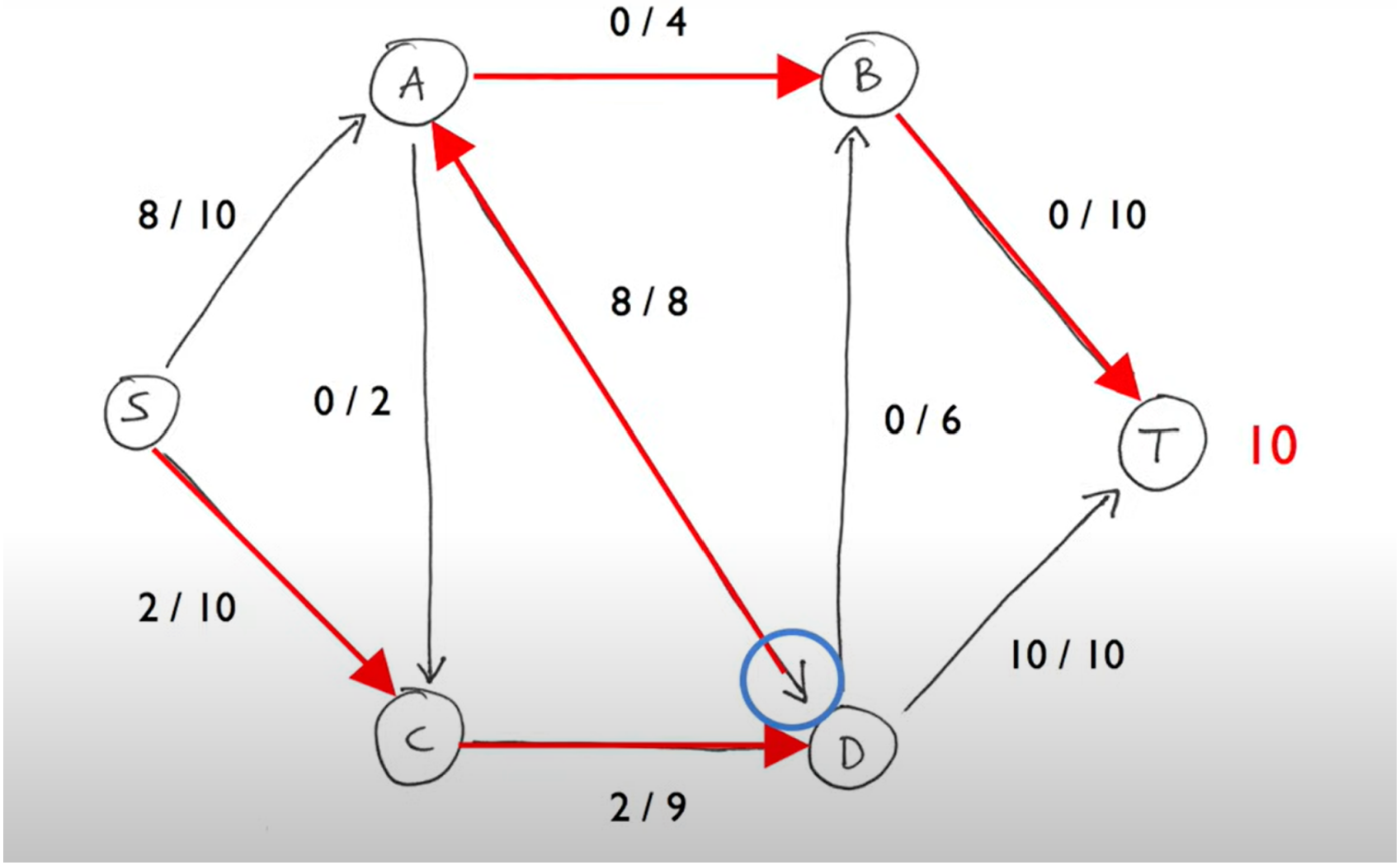
Cited from Michael Sambol

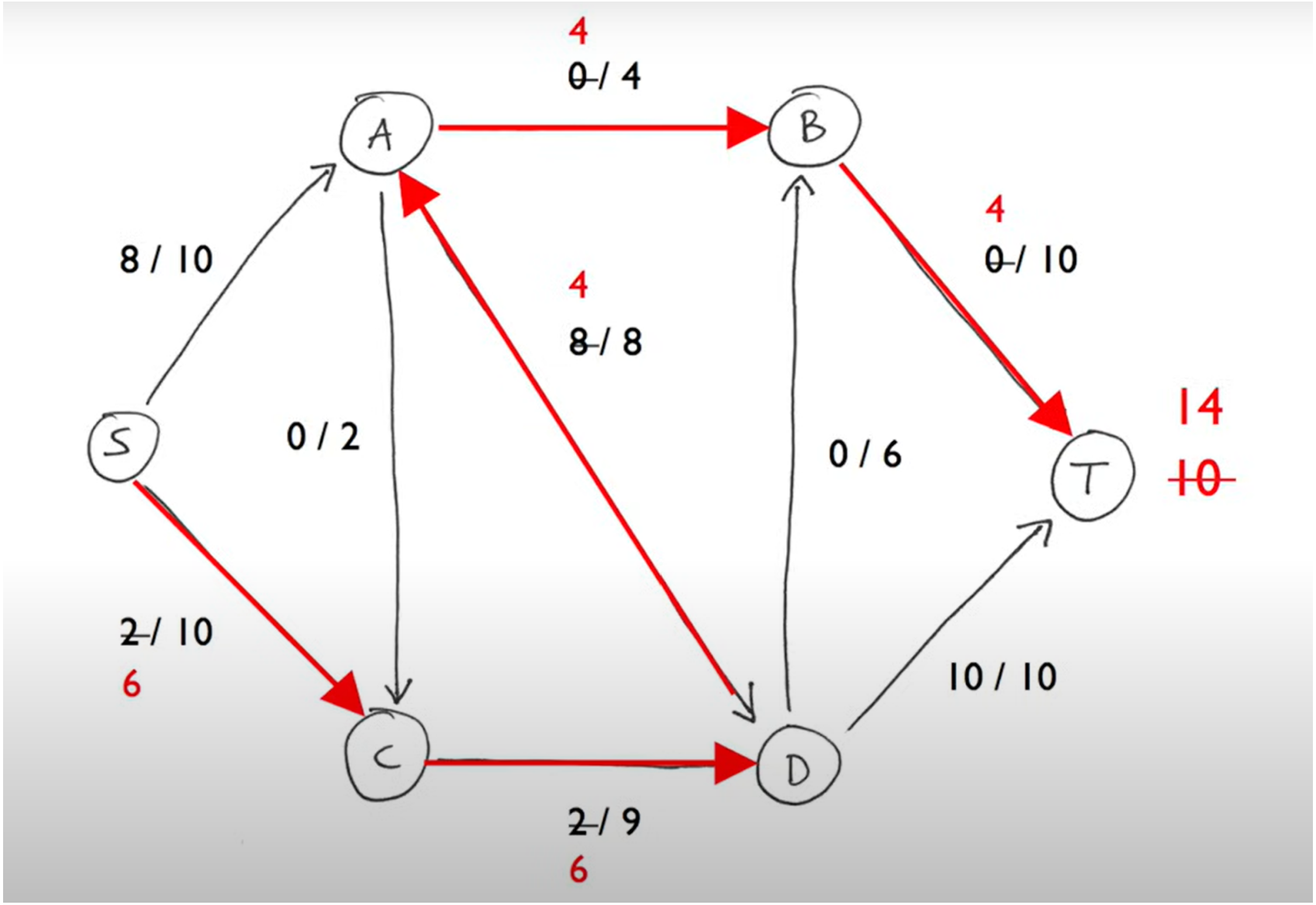

0 / 4
A
B
0 / 10
0 / 10
0 / 8
S
0 / 2
0 / 6
T
0
0 / 10
0 / 10
C
D
0 / 9


0 / 4
A
B
8
0 / 10
8
0 / 8
0 / 10
S
0 / 2
0 / 6
T
8
0
0 / 10
0 / 10
8
C
D
0 / 9


0 / 4
A
B
8 / 10
0 / 10
8 / 8
S
0 / 2
0 / 6
T
8
0 / 10
8 / 10
C
D
0 / 9

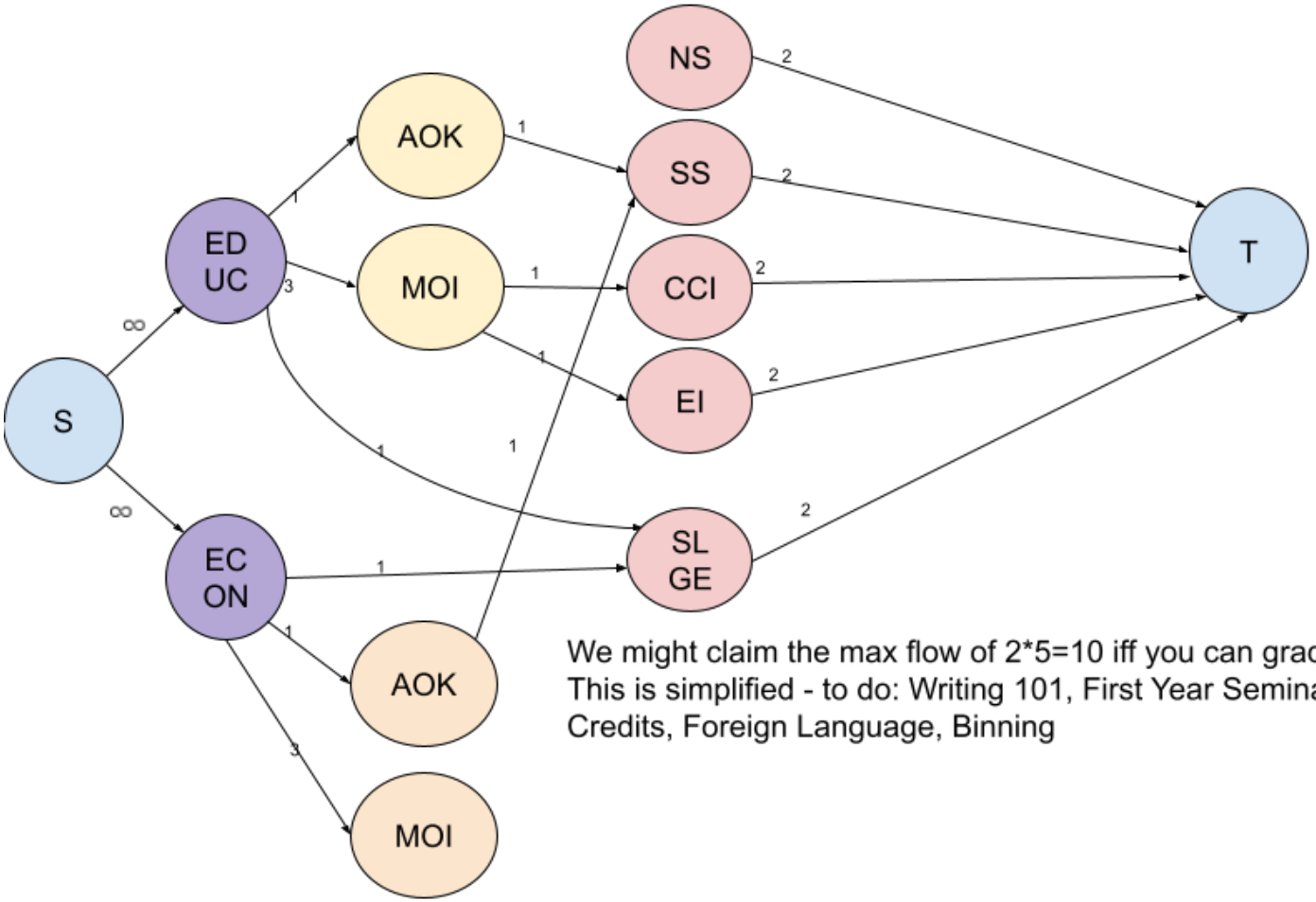

0 / 4
A
B
8 / 10
0 / 10
8 / 8
S
0 / 2
0 / 6
T
10
8
0 / 10
2
8 / 10
10
C
D
0 / 9
2


0 / 4
A
B
8 / 10
0 / 10
8 / 8
S
0 / 2
0 / 6
T
10
2 / 10
10 / 10
C
D
2 / 9



(This is not a max flow, more steps to do but we leave it here). Time complexity: $O(f \cdot E)$ or $O(V \cdot E)$

Architecture

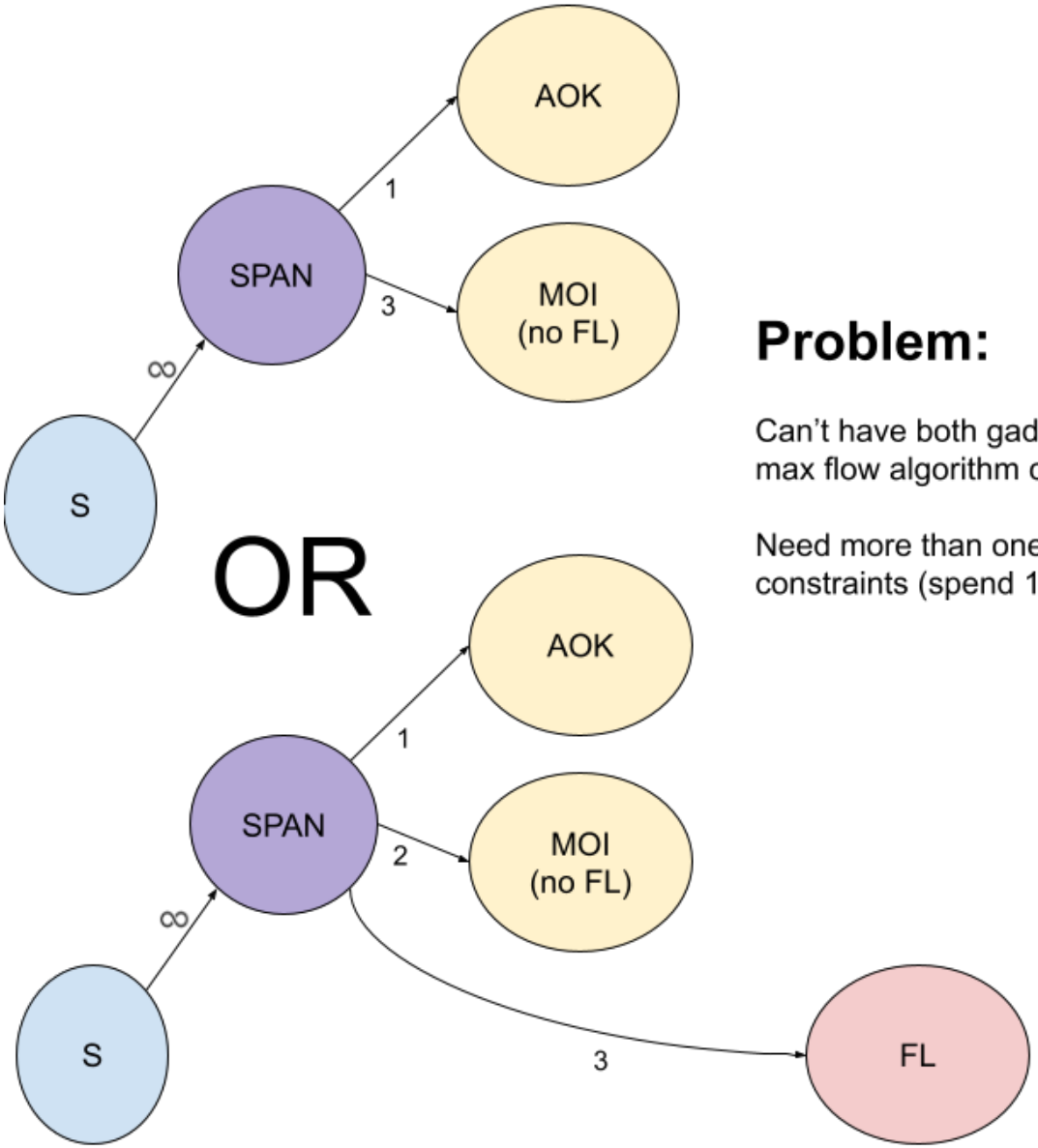


We might claim the max flow of 2*5=10 iff you can graduate. This is simplified - to do: Writing 101, First Year Seminar, ½ Credits, Foreign Language, Binning

Z

Special Constraints

- **Foreign Languages**: Special rules:
  - 3 courses at 100/200 level, or
  - 1 course at 300+ level, which satisfies all 3 requirements.
- Hard to model because of conservation requirements.
- How to model a coupon for a dependent requirement?



Problem:

Can't have both gadgets in the same graph because a max flow algorithm could run partial flows to each gadget

Need more than one gadget because of conservation constraints (spend 1, get 3 free for 300-level FL code)

Binning

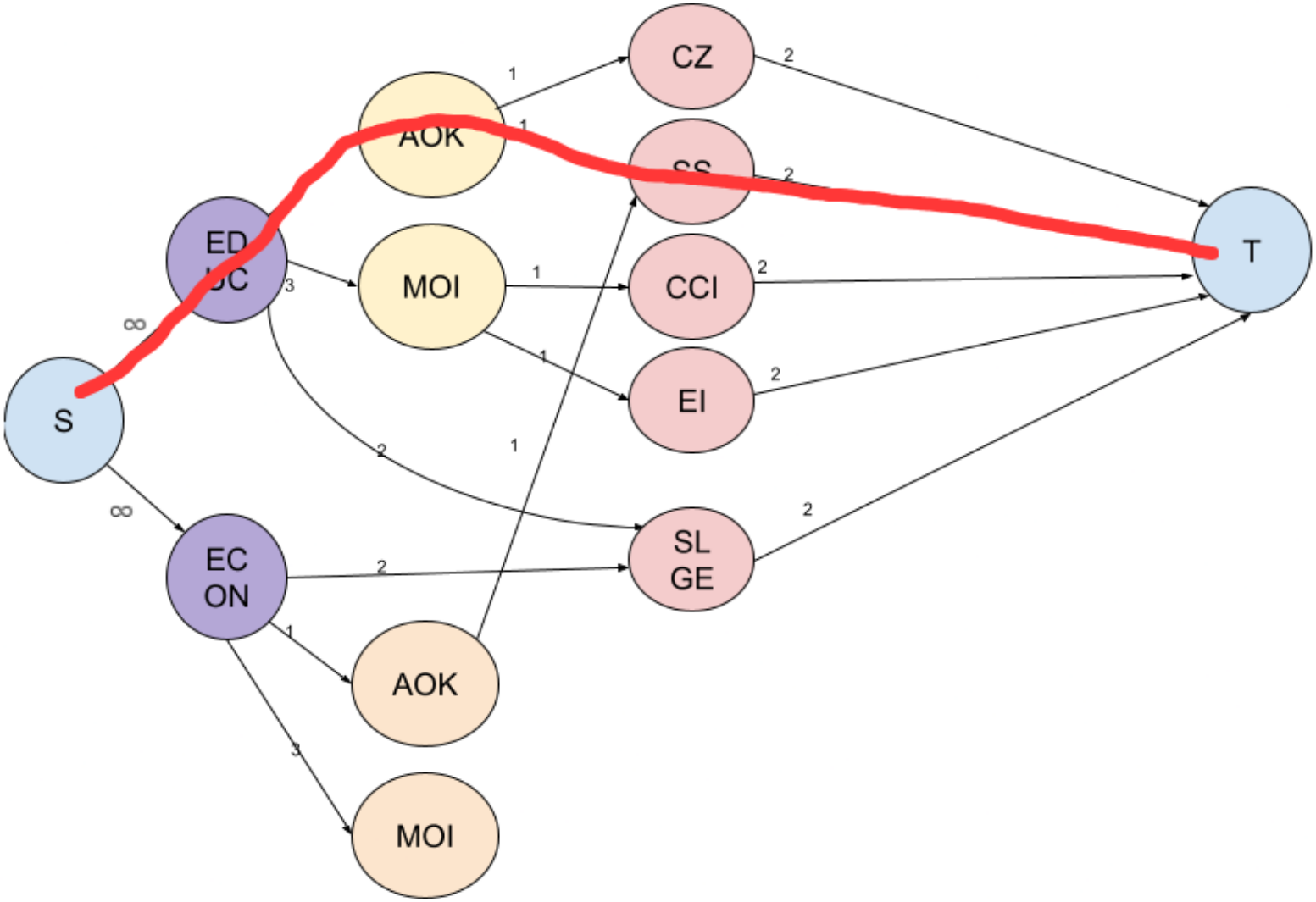
- Many classes share the same codes, increasing the complexity of graph construction and max flow computation.
- Solution: Bin classes into sets of codes, focusing on the sets needed to satisfy requirements.
- A set of codes may be used multiple times (up to the number of available classes).
- Additional upper bounds for code set usage:
  - $2 \times \text{numNonFLcodes} + 3 \times \text{ifFLCode}$
  - 10 (I can construct 10 courses to graduate)
- How many copies? Minimum of all 3 of those bounds.

Handling Already-Taken Classes

- Codes used from already-taken classes are not fixed but can be assigned in multiple ways.
- Assign codes in all possible ways for the given classes:
  - Possibilities for a class: $\binom{4}{3}$ or $\binom{2}{1}$, or rarely their product.
- For each assignment:
  - Build a graph, push flow through the assigned codes.
  - Remove the used flow and reduce capacity accordingly.
  - Decrease the remaining max flow required for graduation.
- Evaluate all graphs and max flows and select the best.

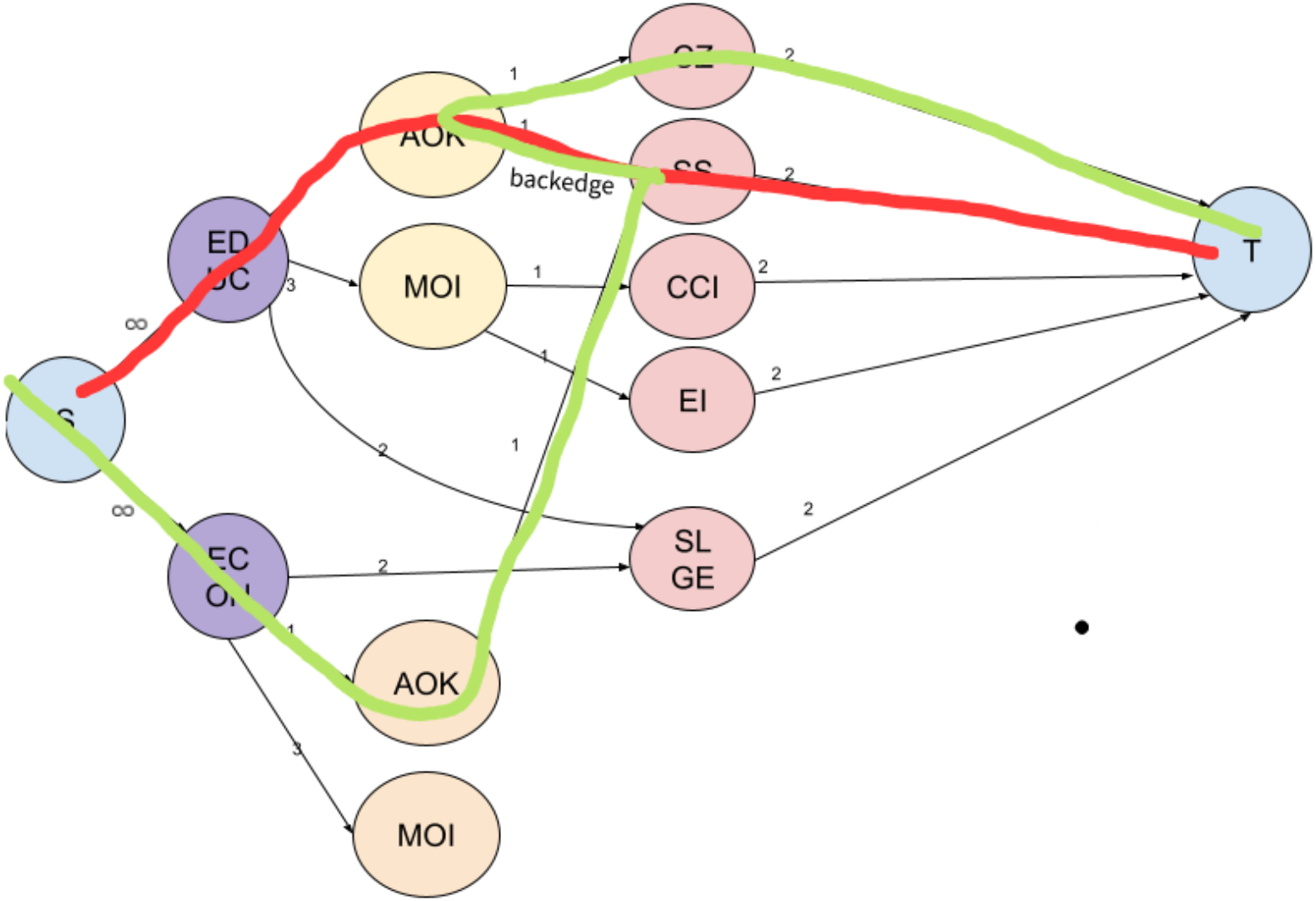
Z

Handling Already-Taken Classes Efficiently



Z

Idea: restrict valid s-to-t paths.

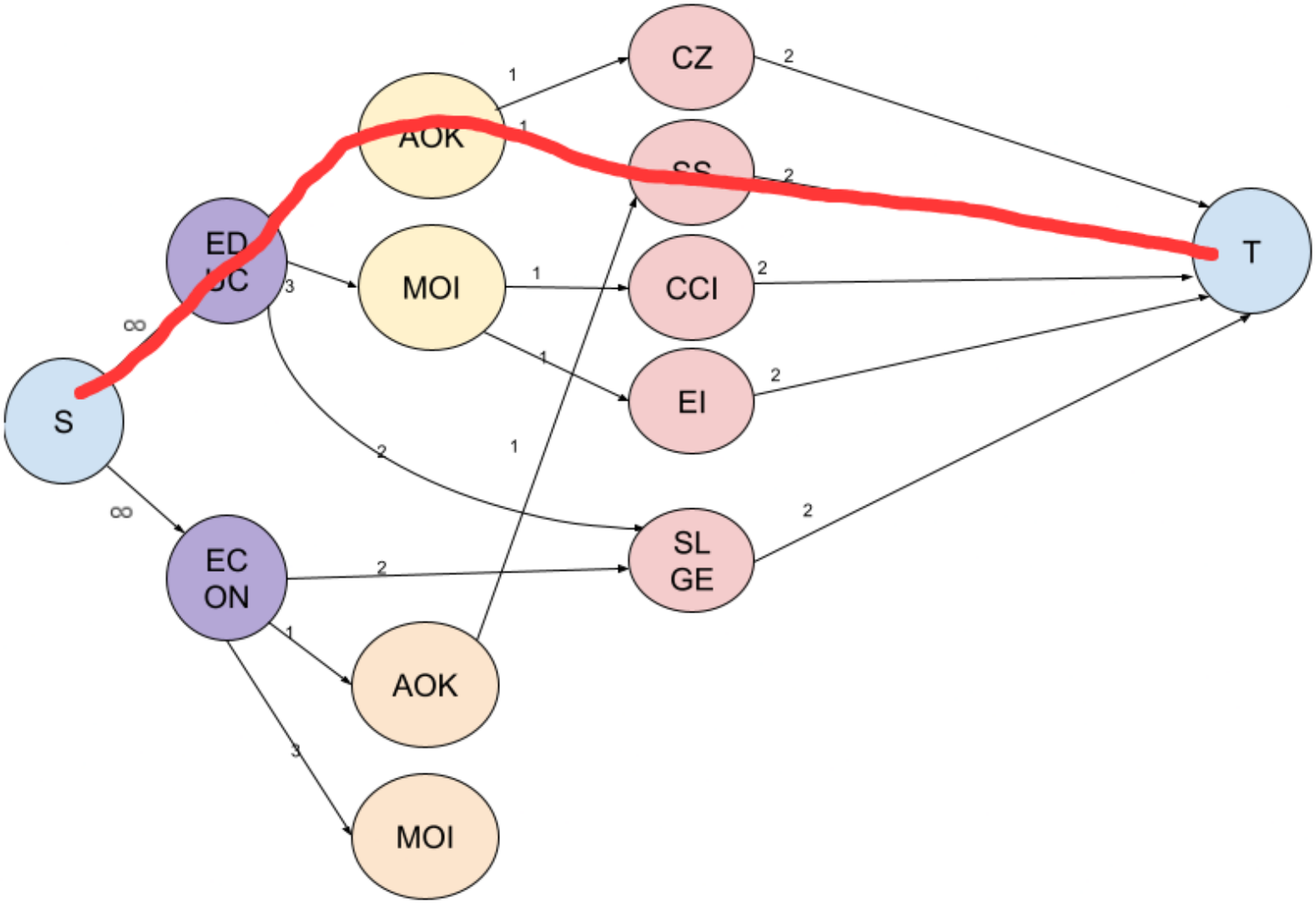


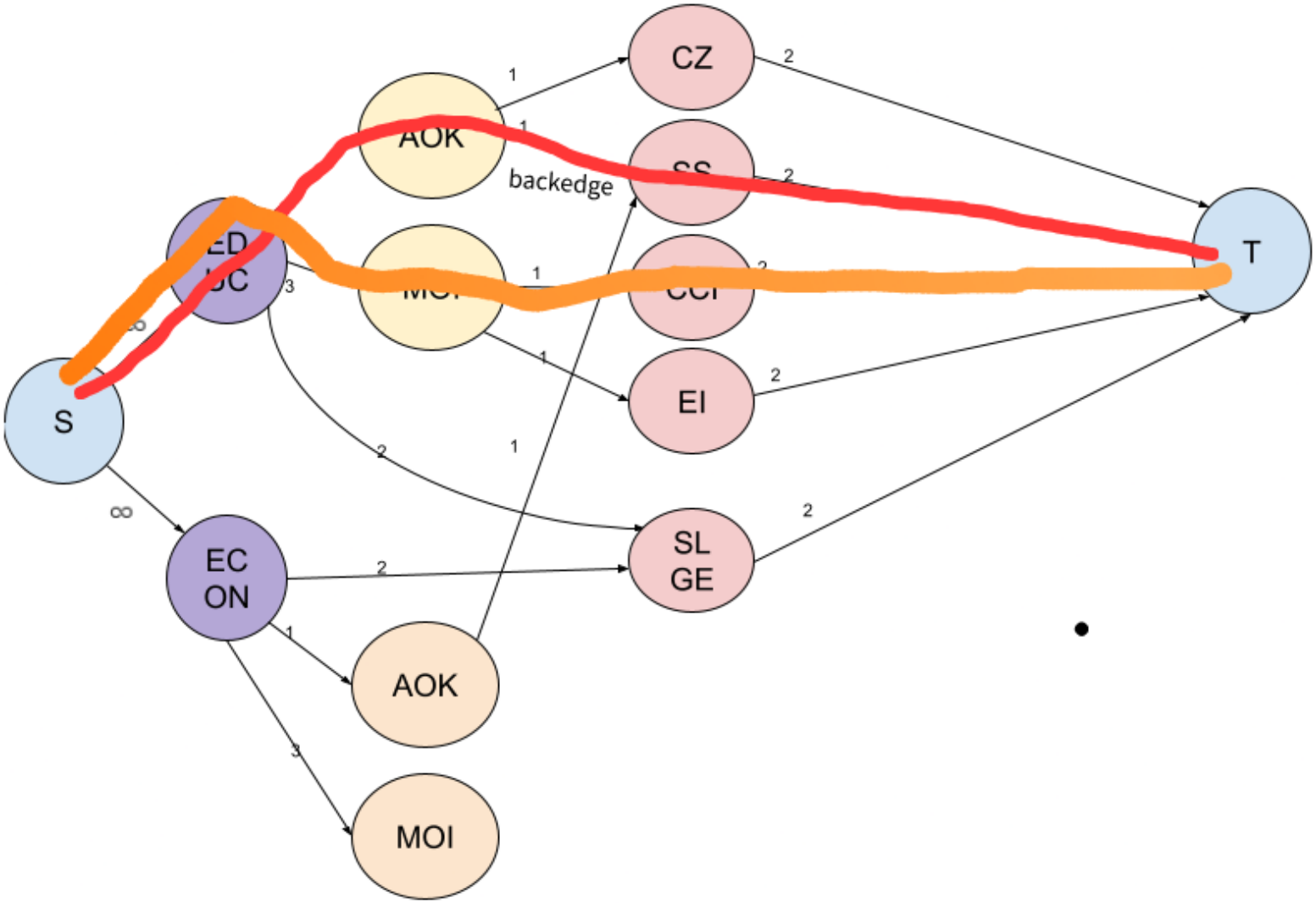
Z

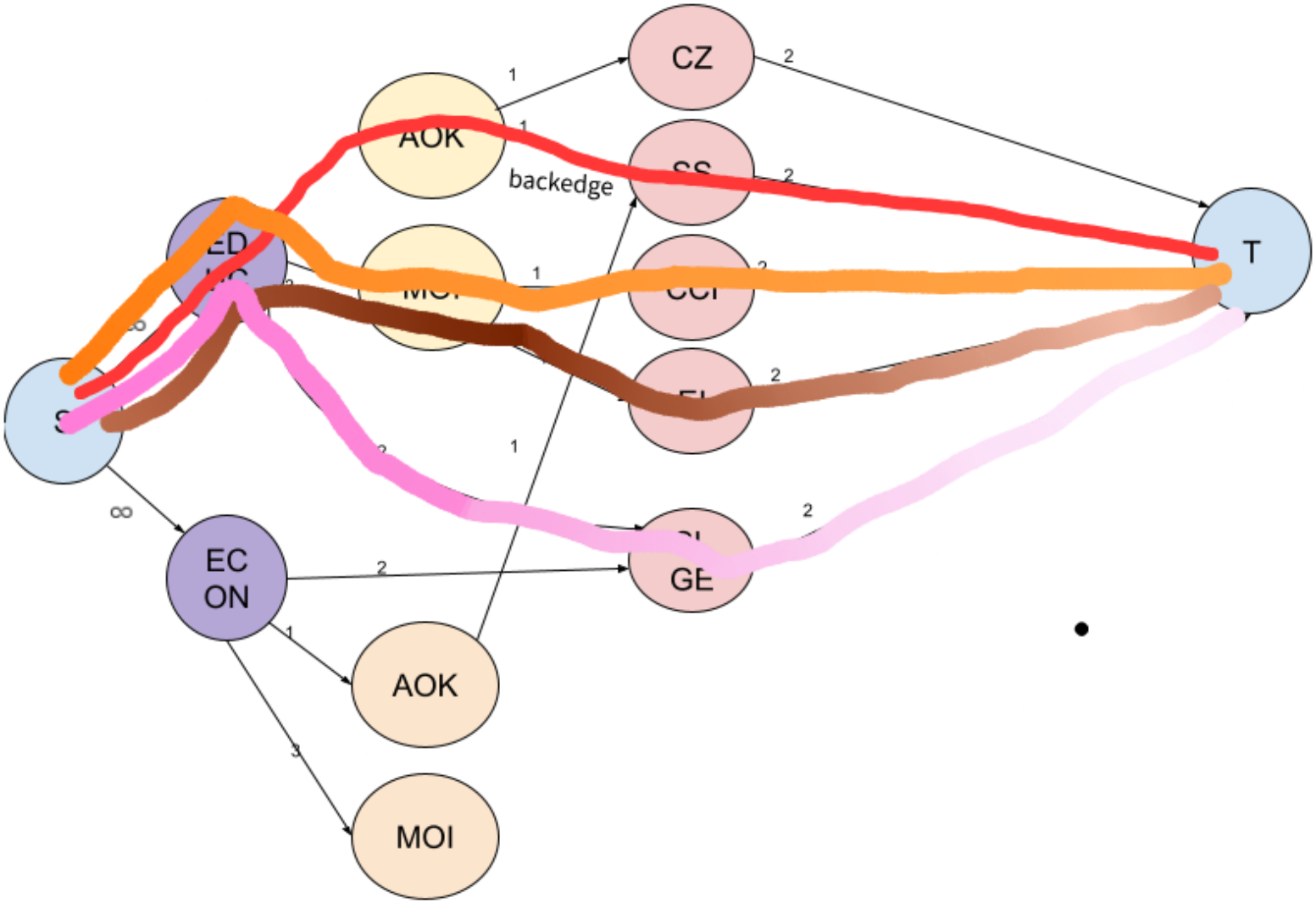
This path is allowed, but backtracking to the course is not.

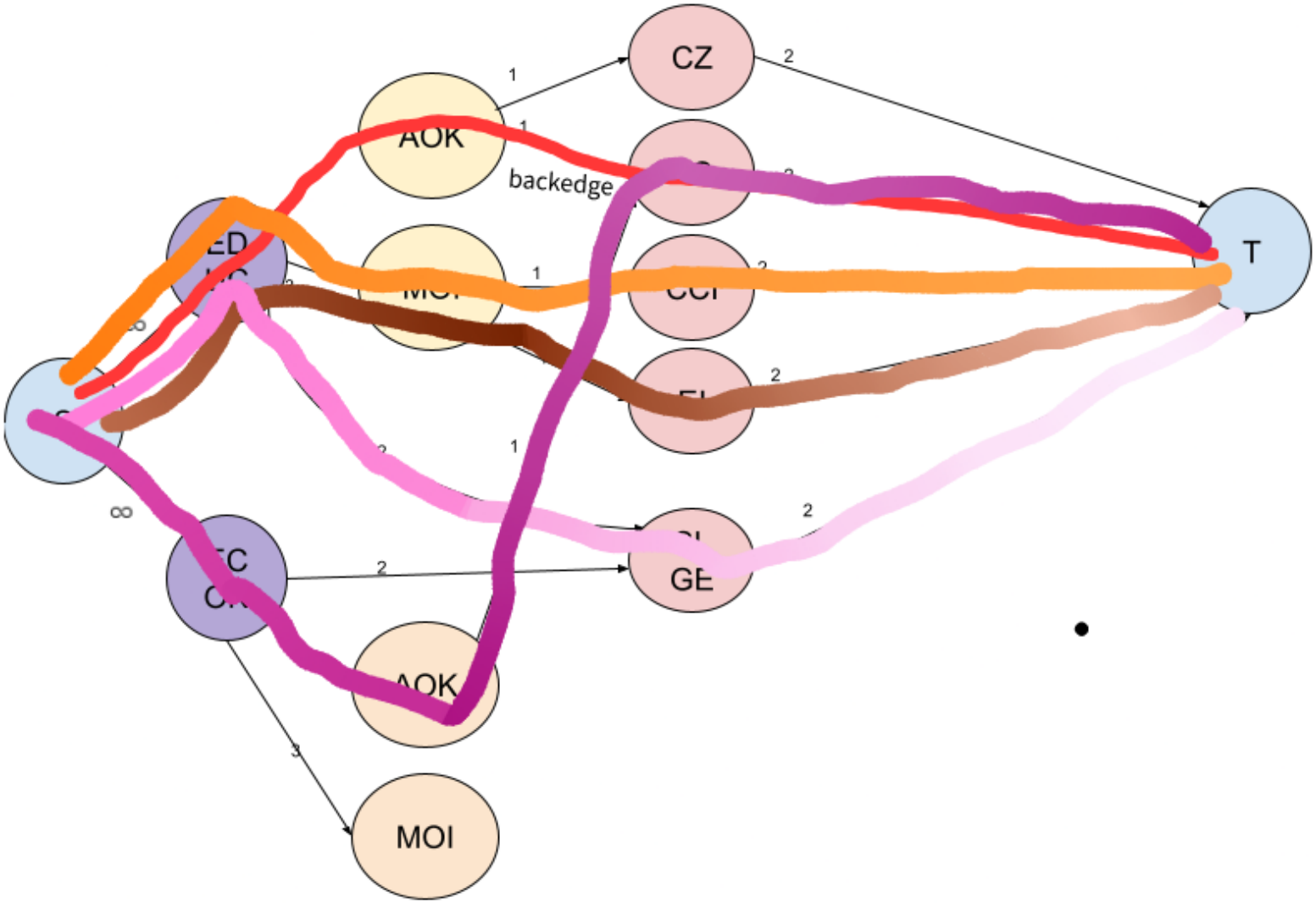
Enumerating All Max Flows

- During max flow computation, at each iteration:
  - Identify all possible s-to-t augmenting paths.
  - For each path, branch into a new state by pushing flow along that path.
- Recursively explore all branches, maintaining flow conservation and capacity constraints.
- Keep only the unique max flow distributions.
- Note: smartly choosing s-to-t paths can improve efficiency. For example, when there is still a path to get more codes on a class already locked in, don't explore new classes.


CZ
AOK
SS
ED
UC
MOI
CCI
T
EI
S
SL
GE
EC
ON
AOK
MOI
1
2
1
2
3
1
2
1
2
∞
1
2
∞
2
1
2
3


CZ
AOK
backedge
T
S
EI
SL
GE
EC
ON
AOK
MOI


CZ
AOK
backedge
T
EC
ON
GE
AOK
MOI
1
2
∞

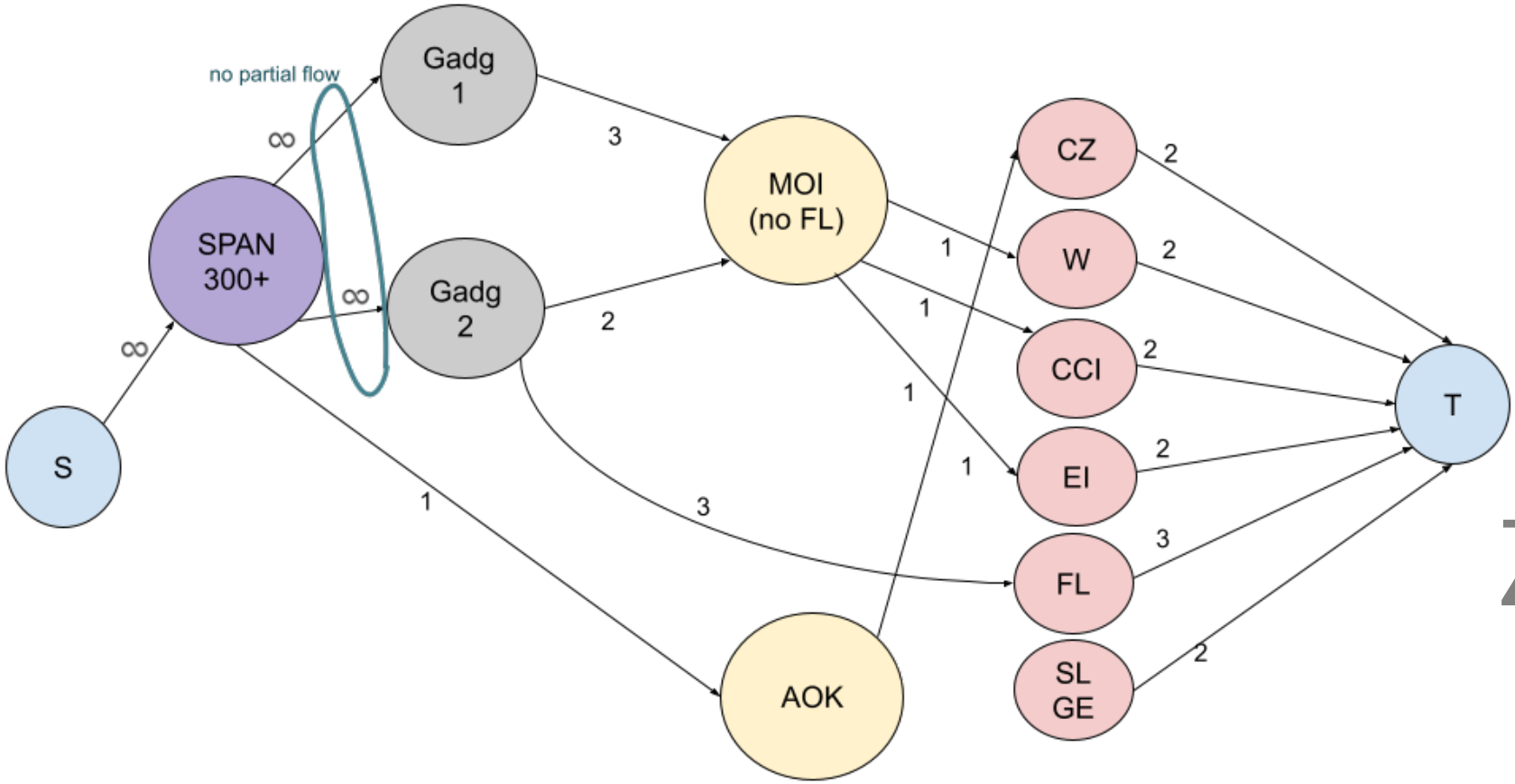

CZ
AOK
backedge
T
GE
AOK
MOI
1
2
∞
3

Finding Max Flows with Minimum Classes

- Goal: Identify maximum flows that use the smallest number of classes.
- If we compute all the max flows then we just need to filter out the ones using more than the minimum number of classes.
- This approach is computational, but there is no easy way to find all max flows subject to a constraint without it (that we can come up with).

Revisiting Foreign Language Gadgets

- **Key Insight:** If we compute all max flows, we can handle the Foreign Language (FL) constraints flexibly.
- **Strategy for 300 Level $\leq$ 2nd class:**
  - Include both FL gadgets:
    - Capacity 3 with no FL option.
    - Capacity 2 MOI no FL with Capacity 3 FL.
- **Process:**
  1. Compute all possible max flows.
  2. Filter out flows that:
     - Send partial flow to both gadgets.
  3. Minimize objectives or return among the valid flows.



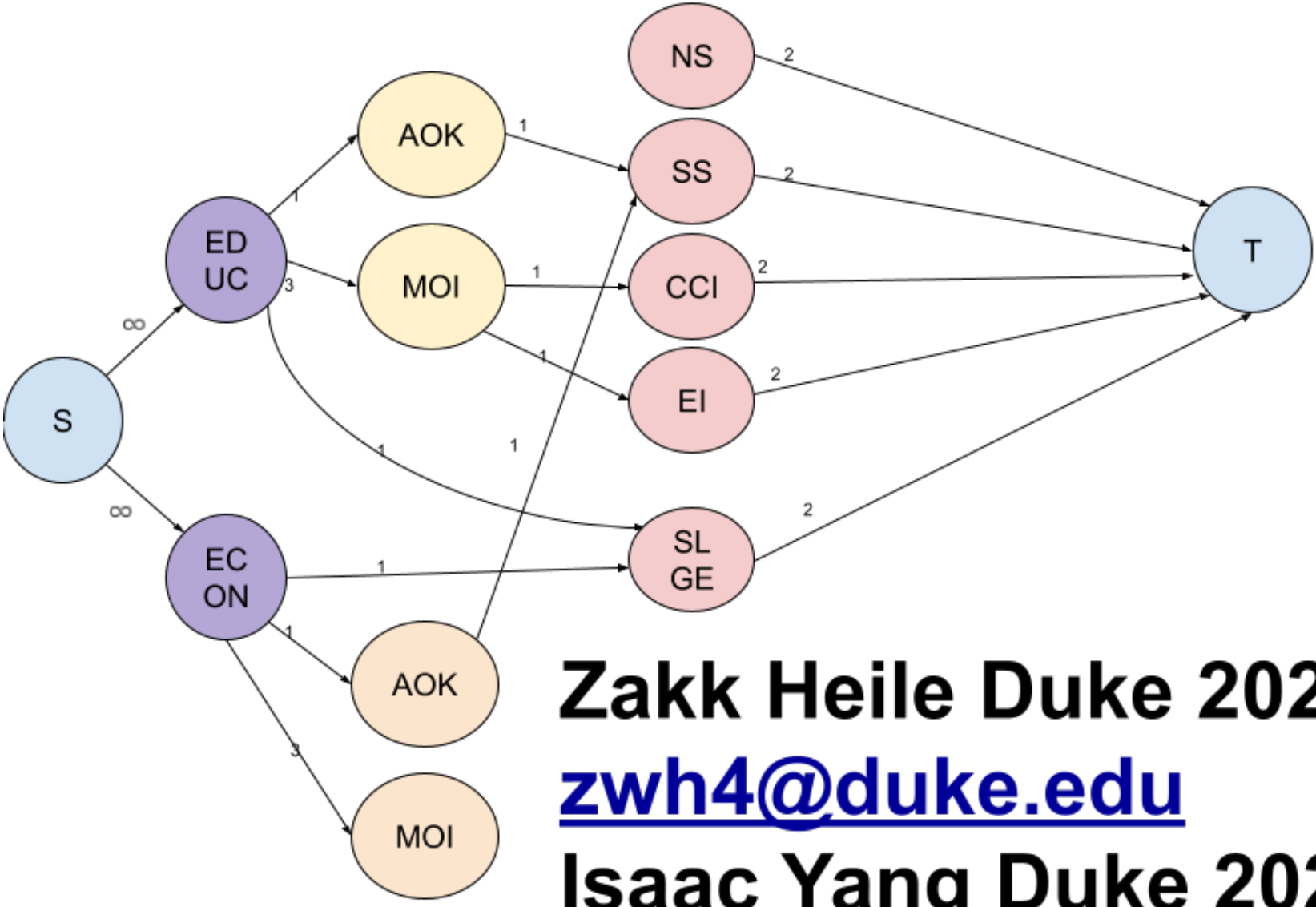
We want to allow backtracking and switching between gadgets, unlike taken courses.

The Power of Graph Modeling

Process data, build graph, and get max flow in $<$3 seconds
with data from $>$9000 courses

Example of Min Cost Max Flow:

- ARTHIST 227: Medieval Castles of Europe - Codes: CZ, STS
- ARTHIST 315: Mapping History with Geographic Information Systems - Codes: CZ, STS
- COMPSCI 321: Graph Analysis with Matrix Computation - Codes: QS
- VMS 325L: Optics and Photonics - Codes: NS
- CULANTH 323: Fundamentals of Global Mental Health - Codes: R, SS
- EVANTH 257: Ecology and Adaptation of Hunters and Gatherers - Codes: CCI, NS
- MATH 431: Introduction to Real Analysis - Codes: QS, W
- PSY 510S: Developmental Psychopathology - Codes: EI, R, SGLE, SS, STS
- RUSSIAN 551: Classics of Russian Literature and Textual Culture - Codes: ALP, CCI, FL, W
- RUSSIAN 552: Russian Culture through Literature - Codes: ALP, CCI, FL, W



Zakk Heile Duke 2027

zwh4@duke.edu

Isaac Yang Duke 2026

iy27@duke.edu