15-780: Graduate AI

Reductions, max-flow

Geoff Gordon (these slides)
Ziv Bar-Joseph
TAs Michael Benisch, Yang Gu

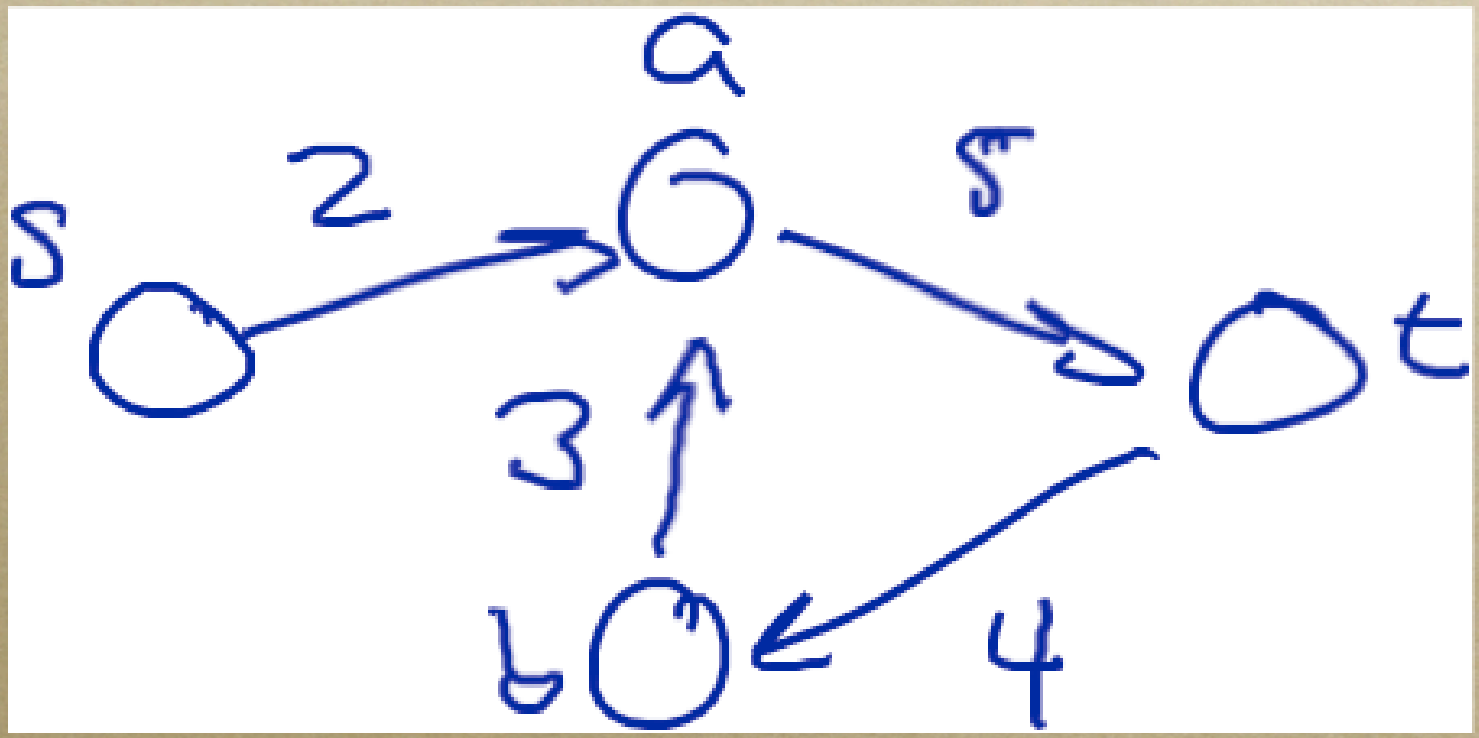
HW2

- *Most people did fairly well on most questions on HW2, although average scores were lower than HW1*
- *Two questions seemed to give lots of people problems:*
  - *1e (reductions)*
  - *3c (max-flow)*

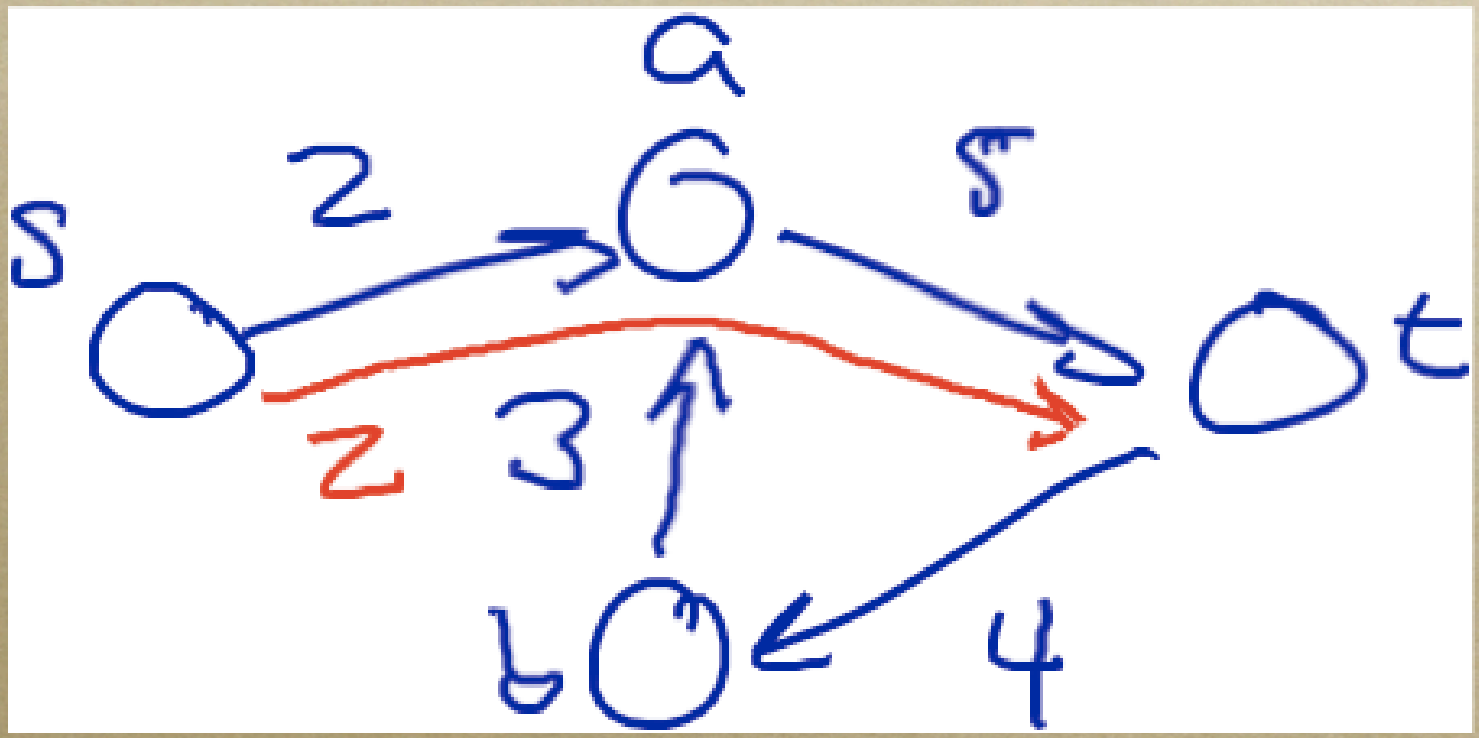
Max-flow

- *Graph (V, E), integer capacities c(e), source s, sink t*
- *Edges represent one-directional pipes: we're allowed to push flow of some fluid (e.g., liters/sec of water) along each*
- *How much flow can we push from s to t?*

Max-flow example



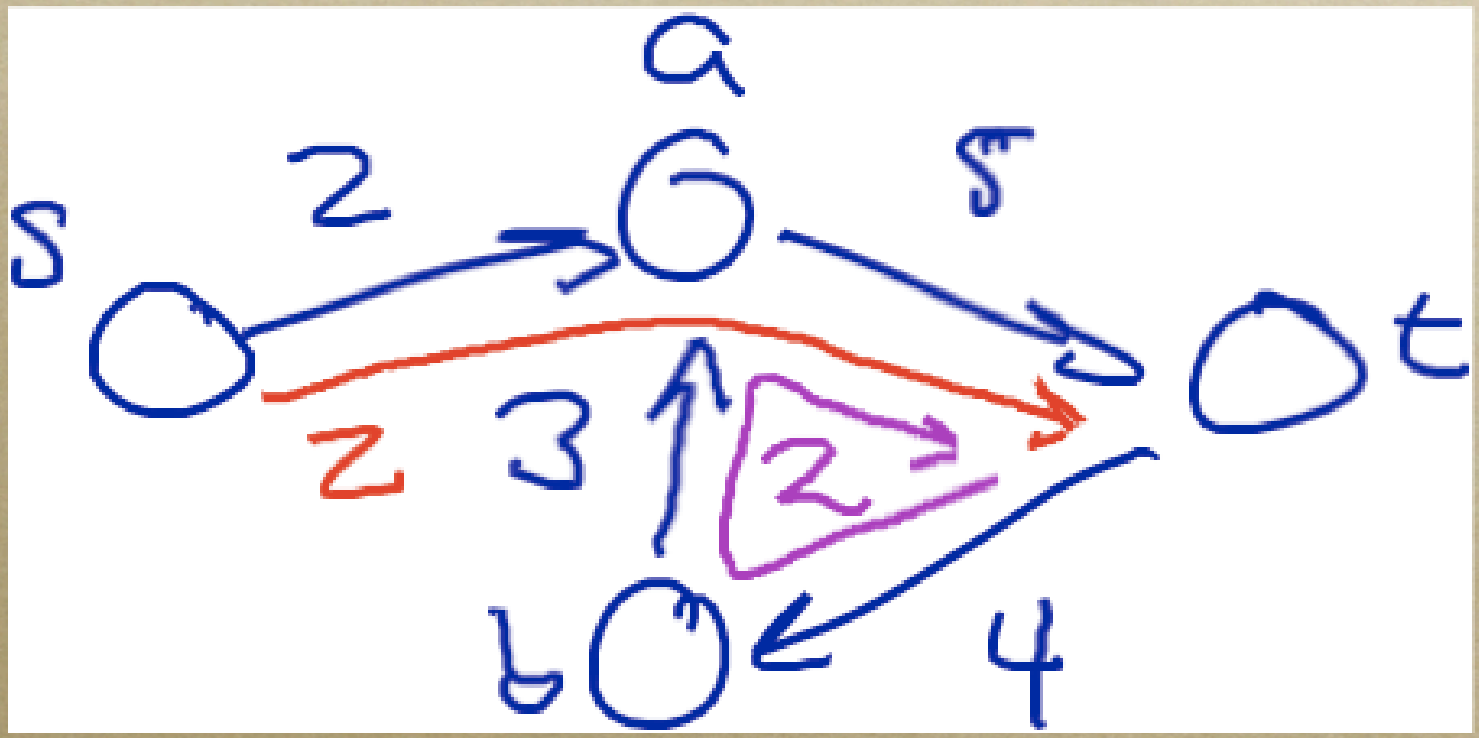
Example flow



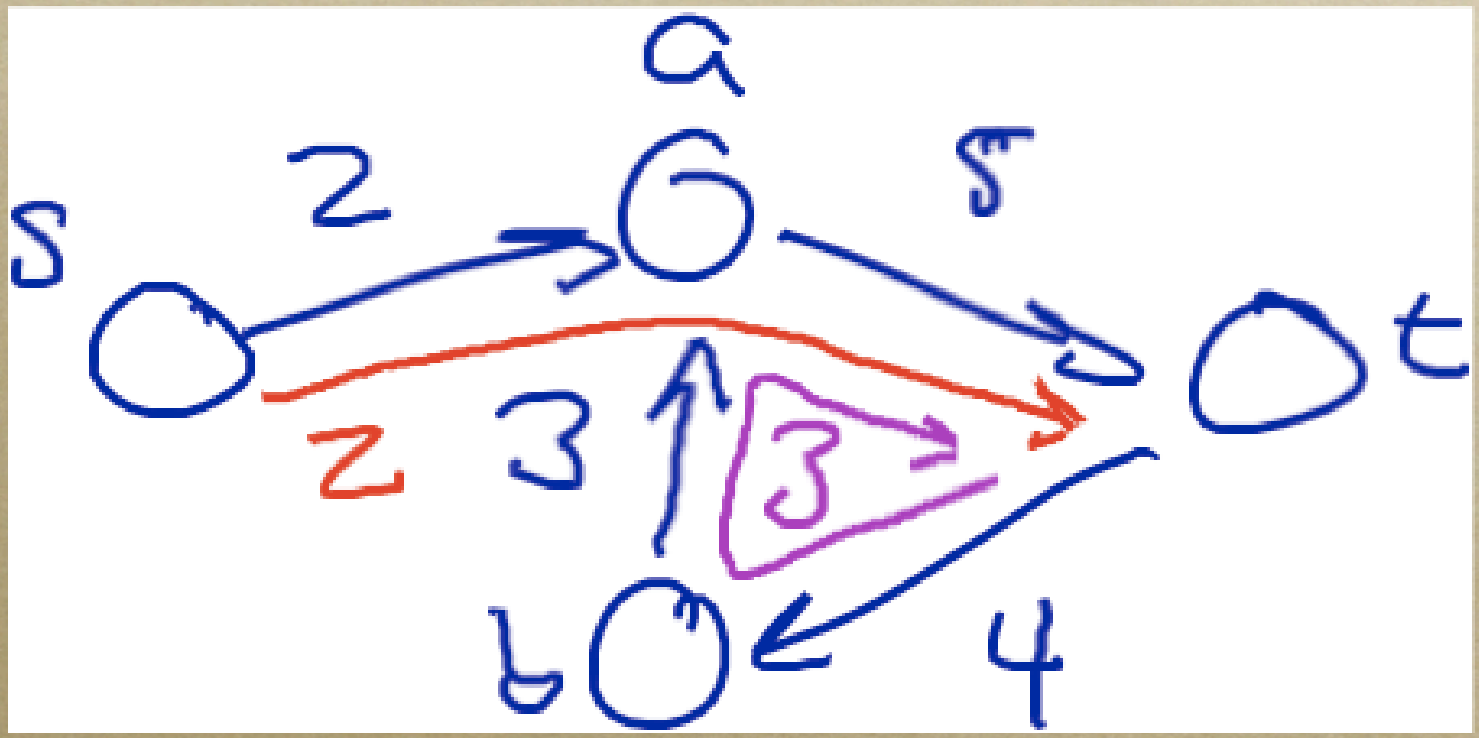
Most common mistake

- *Double-counting flow*
- *If we pump some water into t, then out of t, then back into t, we should only count it once*

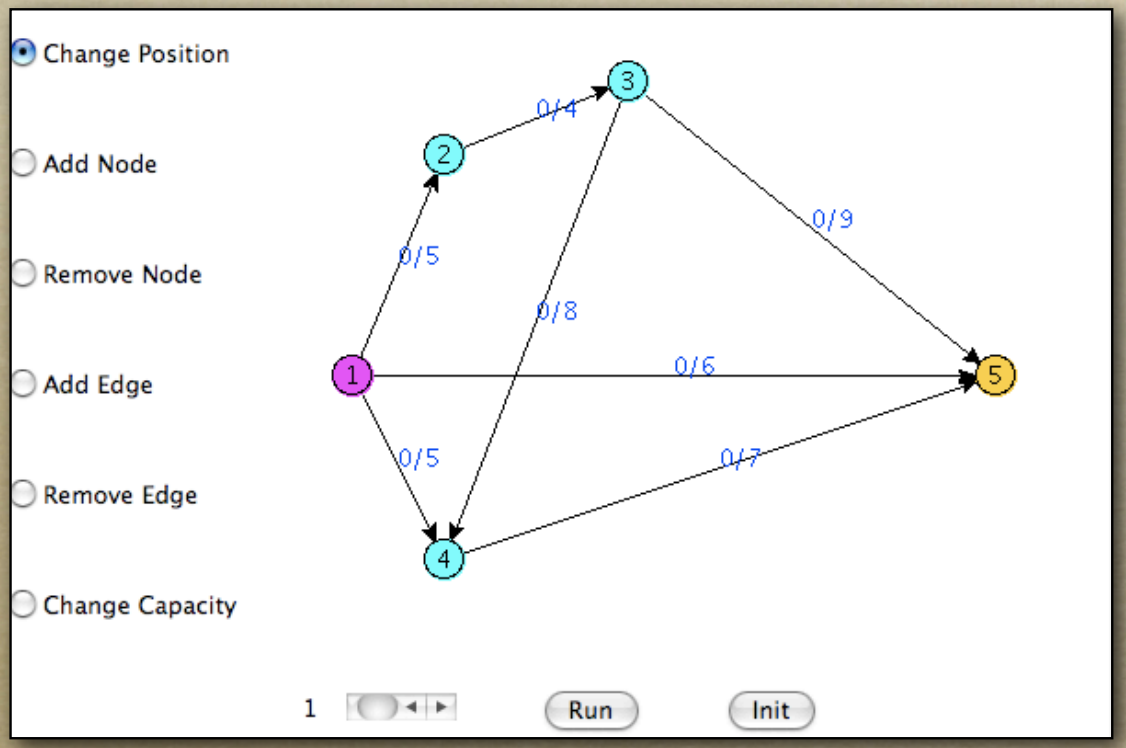
For example



For example



MAXFLOW link



- *http://www.lix.polytechnique.fr/~durr/MaxFlow/*

Decision problems

- *Decision problem = mapping from **instances** (strings) to **decisions** (T/F)*
- *E.g., SAT:*
  - *instance = formula of propositional logic, represented as string*

    $(x \vee y) \wedge \neg x$

  - *decision = T iff formula is satisfiable*

Complexity classes

- *Complexity class = a set of decision problems that can be solved inside some resource limit*
- *Limit may depend on size of instance = length of string encoding it*

Optimization problems

- *Standard complexity classes refer **only** to decision problems*
- *Optimization problems must be translated*
- *E.g., MAXFLOW:*
  - *In addition to usual input, integer target flow f*
  - *Can we push flow $\geq f$ from s to t?*

Example complexity class: P

- *Problems solvable in polynomial time on a polynomial-size computer*
- *"Polynomial" is in instance size n*
- *E.g., time* $\leq 7n^2 + 3n + 5$*, computer size* $\leq 2n$
- *Don't need to check size, just realize that it can grow with n*

MAXFLOW complexity

- *E.g., MAXFLOW can be solved with the Ford-Fulkerson algorithm*
- *At most VE iterations, each taking time* $\leq aE + b$ *(for some a, b)*
- *Instance size n:* $V \leq n$, $E \leq n$
- *So, total time* $\leq n^2(an + b)$

More examples: NP, PSPACE

- *PSPACE = problems that can be solved on a poly-sized computer (in any amount of time)*
- *NP = problems with a poly-sized* ***certificate*** *that allows us to verify in poly-time that we should answer T*
  - *don't have to be able to find F certs*

SAT

- *Typical problem in NP: SAT*
- *Certificate = satisfying assignment*
- *Given assignment, can just substitute it into formula and check that it's correct*
- *Size of cert is linear in n, time to check is also linear*

Nesting

- *Some complexity classes contain others*
- *E.g., $P \subseteq NP \subseteq PSPACE$*
- *So, MAXFLOW $\in PSPACE$*
- *Many other problems in PSPACE are probably much harder than MAXFLOW…*

Worst case complexity

- *Complexity classes measure the **worst-case** complexity of a problem*
- *Most instances of a given size might be easy—but it's the hard ones that determine the complexity class*
- *E.g., easy SAT formulas*
  - *x* ∨ *(formula of size n with no x in it)*

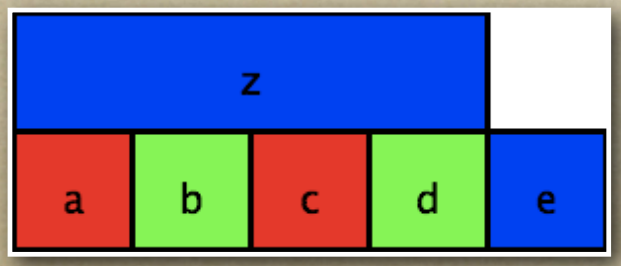
Determining complexity

- *To find the complexity of a problem class, use **reductions***
- *E.g., 1(e) asked for a reduction from planning to SAT*

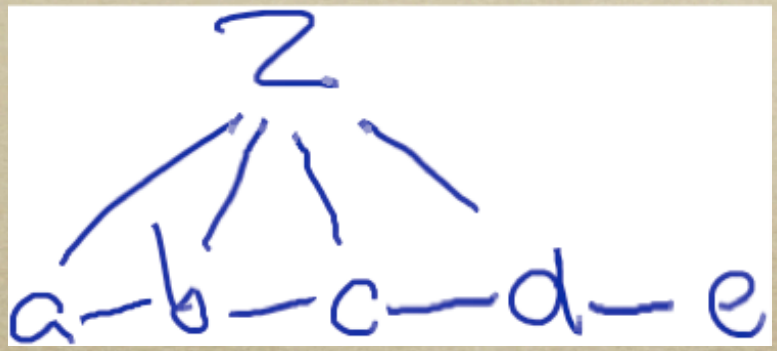
Reduction

- *Loosely, "problem A reduces to problem B" means that if we can solve B then we can solve A*
- *More formally, A, B are decision problems (instances $\mapsto$ truth values)*
- $\exists$ *a poly-time function f so that: given an instance a of A, f(a) is an instance of B, and A(a) = B(f(a))*

Example reduction



=



$\Rightarrow$

$$a_r \vee a_g \vee a_b$$
$$\neg (a_r \wedge y_r)$$
$$\neg (a_r \wedge z_r)$$

Example reduction #2

- *Given a planning problem (like the one in problem 1)*
- *Build plan graph*
- *Translate plan graph to SAT instance*
- *Plan is possible iff formula is satisfiable*

Direction of reduction

- *If A reduces to B then*
  - *if we can solve B, we can solve A*
  - *so B must be at least as hard as A*
    - *ignoring time to compute translation f(a), possible blowup in instance size*
- *Very easy to get direction wrong!*
  - *A does not have to be as hard as B*

Hardness of PLAN

- *So, for 1e: we have shown that SAT is at least as hard as PLAN*
  - *so, PLAN $\in$ NP*
- *And, we have a way to get certificates for PLAN*
  - *satisfying assignment lets us check efficiently that our plan works*