Learning Goals

- Reduce one problem to another ✓
- Define Polynomial Time Reduction ✓
- Describe why reductions are important ✓

Announcements

- No office hours Friday (Fall break)
- Quiz, RD due Mon, PS due Thurs (Fall break)

Exit Tickets

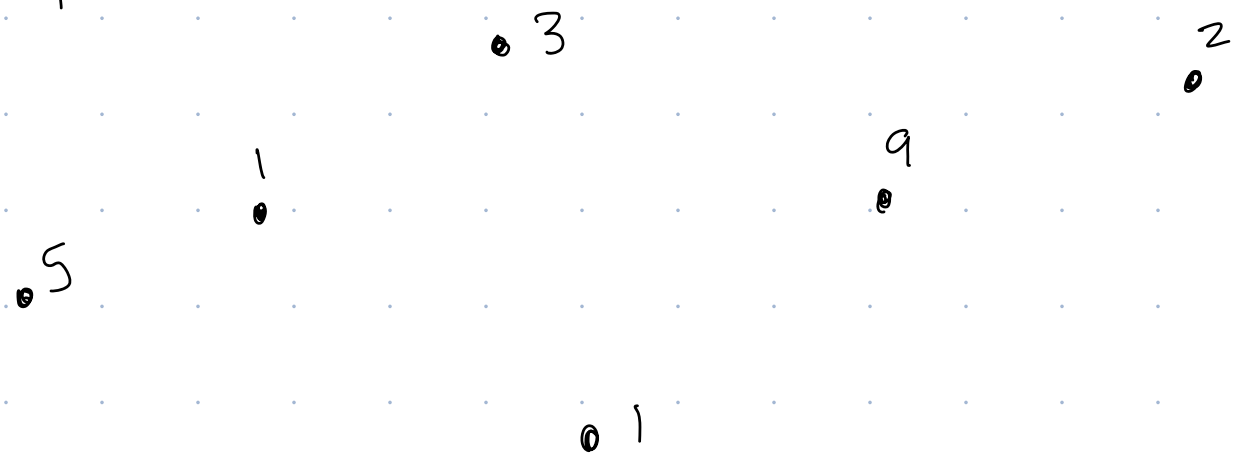
Why "dynamic programming"?
Which strategy when?

≥2 vertices?
Always need to work backwards?
Tree MWIS? Other graphs? Other non-graph problems?
While loop?
Recursion in dynamic prog.?
Store vertices?

Cell Tower Scheduling

(latitude, longitude)

Input: Array P s.t. P[i] is the location of i^{th} cell tower.
Array D s.t. D[i] is the number of data packets to send from i^{th} tower.

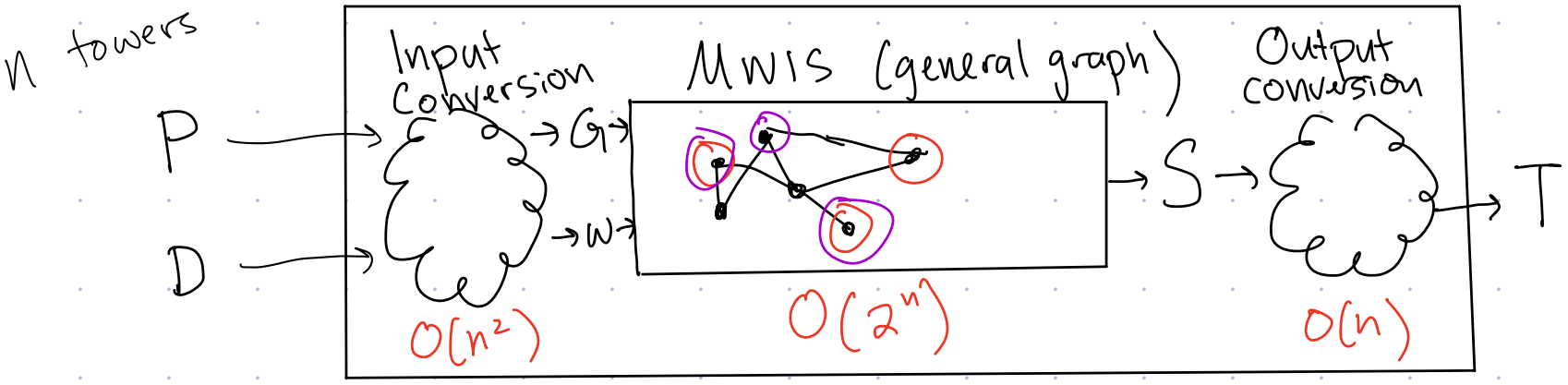


Output: Set of towers T to broadcast in next time step.

- If 2 towers within 2 miles of each other broadcast at same time → interference
- Maximize the number of data packets that are in queues that reduce in length.

Reduction

Cell Tower Problem



1. What should input/output conversion functions be?

I.C. (P, D) $O(n^2)$

$V \leftarrow \{1, 2, 3, \ldots n\}$

For $i \in [n]$: $w(i) \leftarrow D[i]$

For each pair $i, j \in [n]$:

If $d(t_i, t_j) \leq 2$, Add $\{i, j\}$ to E

- Return $(G = (V, E), w)$

O.C. (S) $O(n)$

- $T \leftarrow S$
- Return T

2. Ethical Matrix (Stakeholders, Well-Being, Autonomy, Justice)

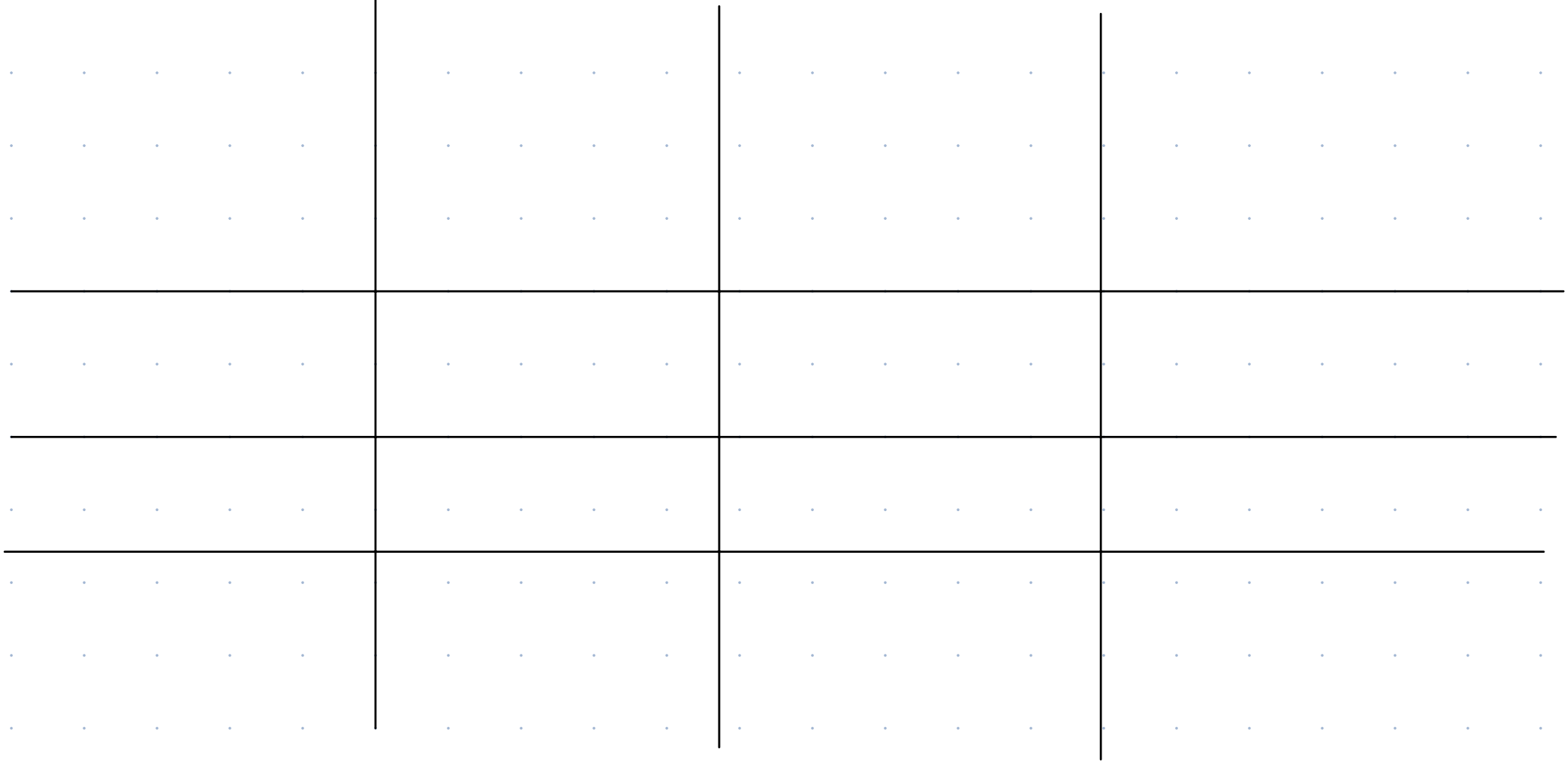
Multiple groups of customers based on how alg treats groups differently

3. Runtime of conversions in terms of n?

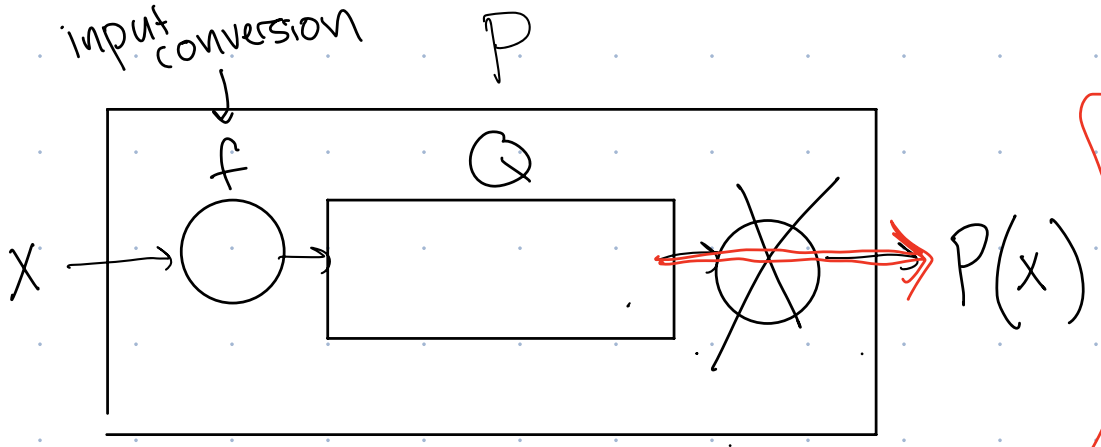
Ethical Matrix (O'Neil + Gunn)

Cell Tower

Stakeholders	Well-Being Harm? Benefit? ↑	Autonomy Choice to use? Are users informed enough to understand meaningfully take responsibility for use? ↑	Justice Unfair treatment of different groups? Access to Tech/Alg?
Cell Company	:) Happy customers Return on investment ^ Competition b/t carriers	✓	Budget, Establishe companies, monopoly?
Rural Users	:(:) small queues not prioritized	X	Lacking tech infrastructure
Urban Users	:(:(large queues :) :) priorities	X	Access to carriers



Polynomial-Time Reduction



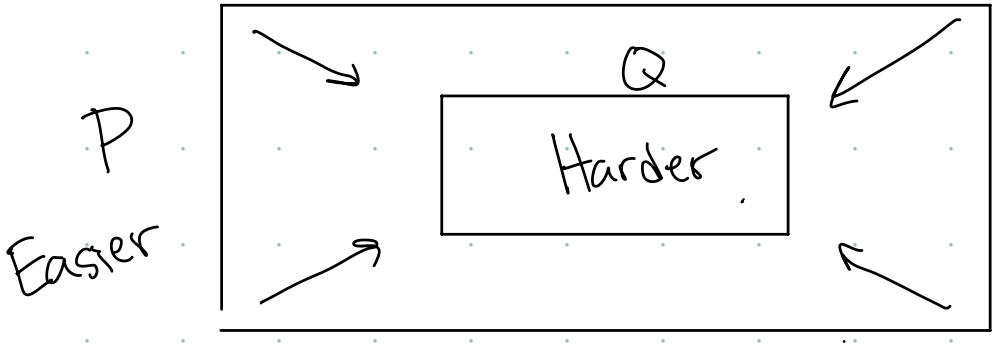
Runtime is $O(n^d)$ for a constant d.

ex: $O(n)$, $O(1)$, $O(n^{10})$

Not allowed: ~~$O(2^n)$~~

def: If you can solve problem P using a solver for problem Q and f runs in polynomial time, then "P is polynomial-time reducible to Q", denoted $P \leq_P Q$

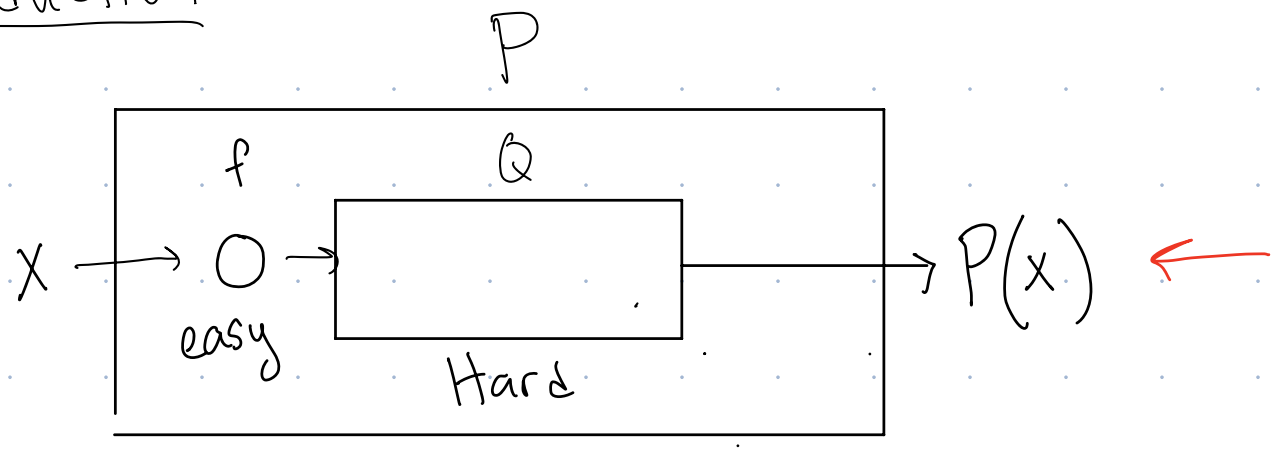
- Q is so powerful, it can not only solve Q, it can also solve P
- Runtime of P is less than that of Q (up to polynomial factors)

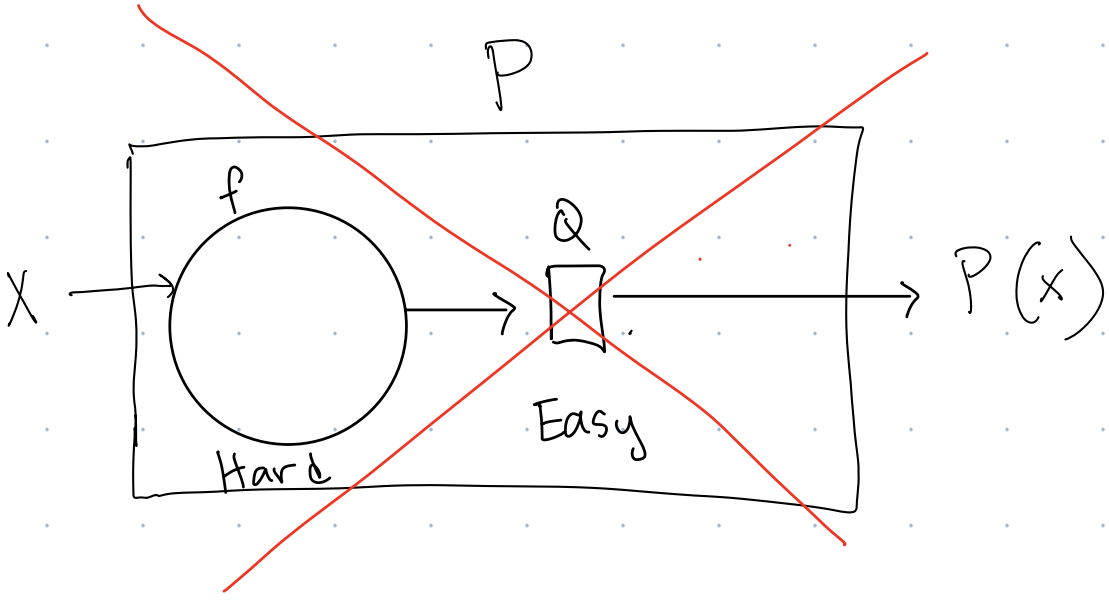


P reduces to Q

(smaller box is harder, more powerful)

Generic Reduction





Why think about reductions?

- Practical: If have an alg for Q, can use it for P

- Conceptual: Gives us a way to compare the difficulty of problems, to compare resources needed to solve problems.