CS265 Class 2

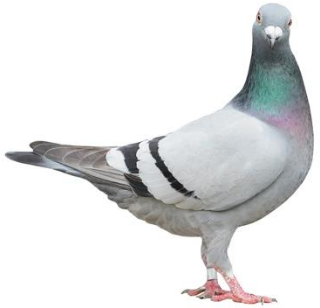
Linearity of Expectation; Coupon Collecting; Karger-Stein

As we get started:

- If you didn’t remember your nametag from last time, make a new one!
- Get started on the “**Warm-Up**” on the Agenda!

Warm-up Question

- There are n pigeons and n pigeon-holes; each pigeon has its own pigeon-hole.
- After a wild night, the n pigeons return to a uniformly random pigeon-hole (so it could be that some holes are empty, and some have more than one pigeon).
- What's the expected number of empty pigeon-holes?



Welcome Back!

- Announcements:
  - HW1 is released! Due a week from Friday. Find it on course website.

Quick recap of mini-lectures

- Linearity of Expectation

$$\mathbb{E}[aX + Y] = a\mathbb{E}[X] + \mathbb{E}[Y]$$

So useful!

- Karger's Algorithm
  - Randomized algorithm for finding min-cuts
  - Basic idea: randomly contract edges until only two "mega-vertices" remain
    - And then repeat a bunch

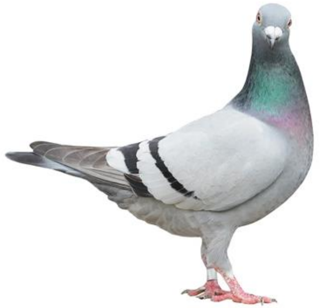
Questions?

Discuss with your group

- Mini-lectures?
- Quiz?

Warm-up Question

- There are n pigeons and n pigeon-holes; each pigeon has its own pigeon-hole.
- After a wild night, the n pigeons return to a uniformly random pigeon-hole (so it could be that some holes are empty, and some have more than one pigeon).
- What's the expected number of empty pigeon-holes?



Warm-Up: E[number of empty pigeon-holes]

About $n/2$

0%

About n/e

0%

$n(1-1/n)^n$

0%

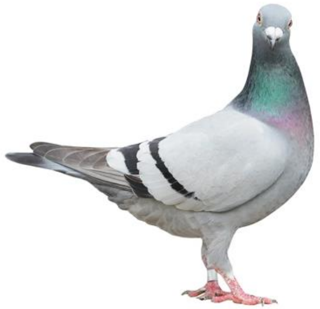
$(n/2)^n$

0%

None of the above

0%

Answer to warm-up



- The answer is $n\left(1-\frac{1}{n}\right)^n \to \frac{n}{e}$ as $n \to \infty$

$$\mathbb{E}\left[\sum_{i=1}^{n} \mathbf{1}[\text{hole } i \text{ is empty}]\right]$$

$$= \sum_{i=1}^{n} \Pr[\text{ hole } i \text{ is empty }]$$

$$= n\left(1-\frac{1}{n}\right)^n \approx n(e^{-n})^n = \frac{n}{e}$$

Coupon Collecting

Coupon Collecting

```
Hello
Randomized
Algorithms
Penguin
Spaceship
Marmoset
…
Mushroom
```

n words

Push the button to get a random word (with replacement)

Coupon Collecting

Hello
Randomized
Algorithms
Penguin
Spaceship
Marmoset
…
Mushroom

n words

Push the button to get a random word (with replacement)

Penguin
Penguin
Randomized
Hello
Randomized
Spaceship
…

X_i is the time where you see your i'th new word.

In this example:

$X_1 = 1$
$X_2 = 3$
$X_3 = 4$
$X_4 = 6$

How many times do you have to push the button to see all the words?

Group Work

X_i is the time where you see your i'th new word.

1. What is $\mathbb{E}X_1$? (This is not a trick question).
2. What is $\mathbb{E}(X_2 - X_1)$? That is, in expectation, how many times do you press the button, after you have seen the first word, before you see a new, second word?
3. What is $\mathbb{E}(X_3 - X_2)$?
4. For any $i = 2, 3, \ldots, n$, what is $\mathbb{E}(X_i - X_{i-1})$?
5. Use your answers to the above, plus linearity of expectation, to answer our question: what is the expected number of times you push the button before you see all n words? It's okay if your answer is a summation, but if you have time try to simplify it to get a big-Theta expression.

Group Work Recap: Questions 1-4

- $\mathbf{E}[X_1] = 1$
- $\mathbf{E}[X_2 - X_1] = \dfrac{1}{1 - \frac{1}{n}}$
- $\mathbf{E}[X_3 - X_2] = \dfrac{1}{1 - \frac{2}{n}}$
- $\mathbf{E}[X_i - X_{i-1}] = \dfrac{1}{1 - \frac{i-1}{n}}$

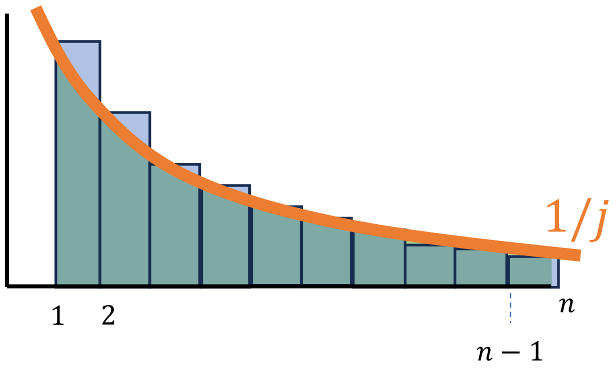
Group Work Recap: Question 5

- $\mathbf{E}[X_n] =$

Group Work Recap: Question 5

- $\mathbf{E}[X_n] = \mathbf{E}[\mathrm{X}_1 + (\mathrm{X}_2 - \mathrm{X}_1) + (\mathrm{X}_3 - \mathrm{X}_2) + \cdots + (X_n - X_{n-1})]$
- $= \mathbf{E}X_1 + \mathbf{E}[X_2 - X_1] + \mathbf{E}[X_3 - X_2] + \cdots + \mathbf{E}[X_n - X_{n-1}]$
- $= 1 + \sum_{j=1}^{n-1} \frac{1}{1-j/n}$
- $= 1 + \sum_{j=1}^{n-1} \frac{n}{j}$
- $= 1 + n \sum_{j=1}^{n-1} \frac{1}{j}$
- $= \Theta(n \log n)$

$$\sum_{j=1}^{n-1} \frac{1}{j} \approx \int_{j=1}^{n} \frac{1}{j}\, dj = \ln(n)$$

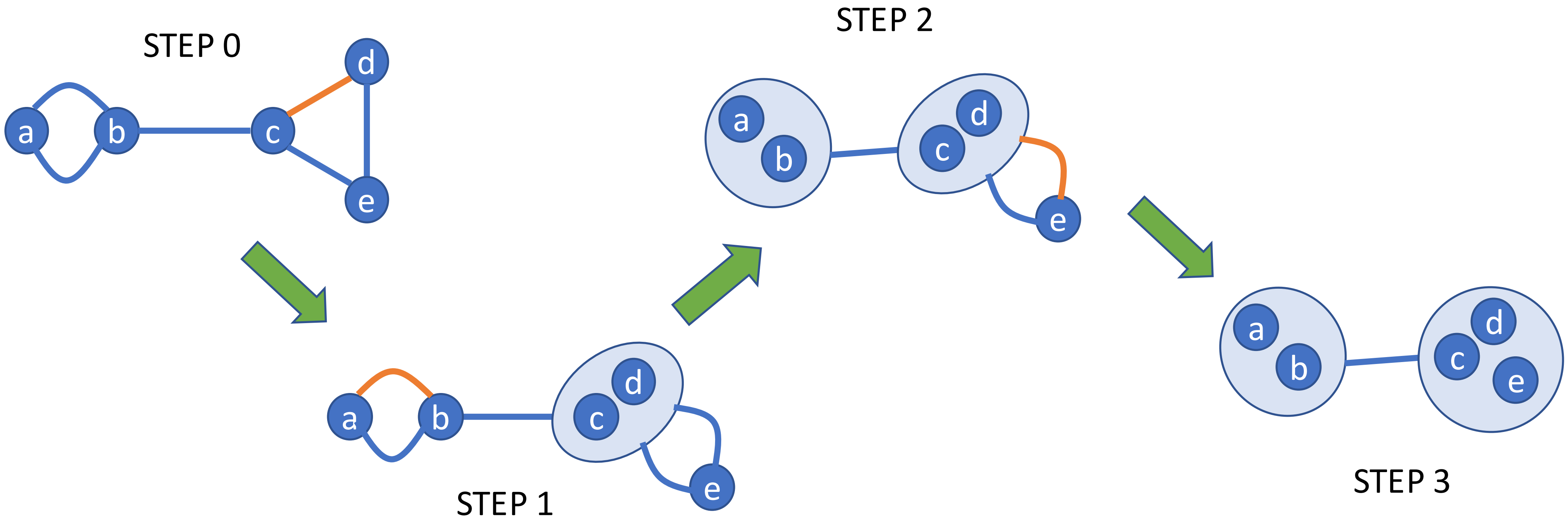


Karger-Stein Algorithm

Karger's algorithm

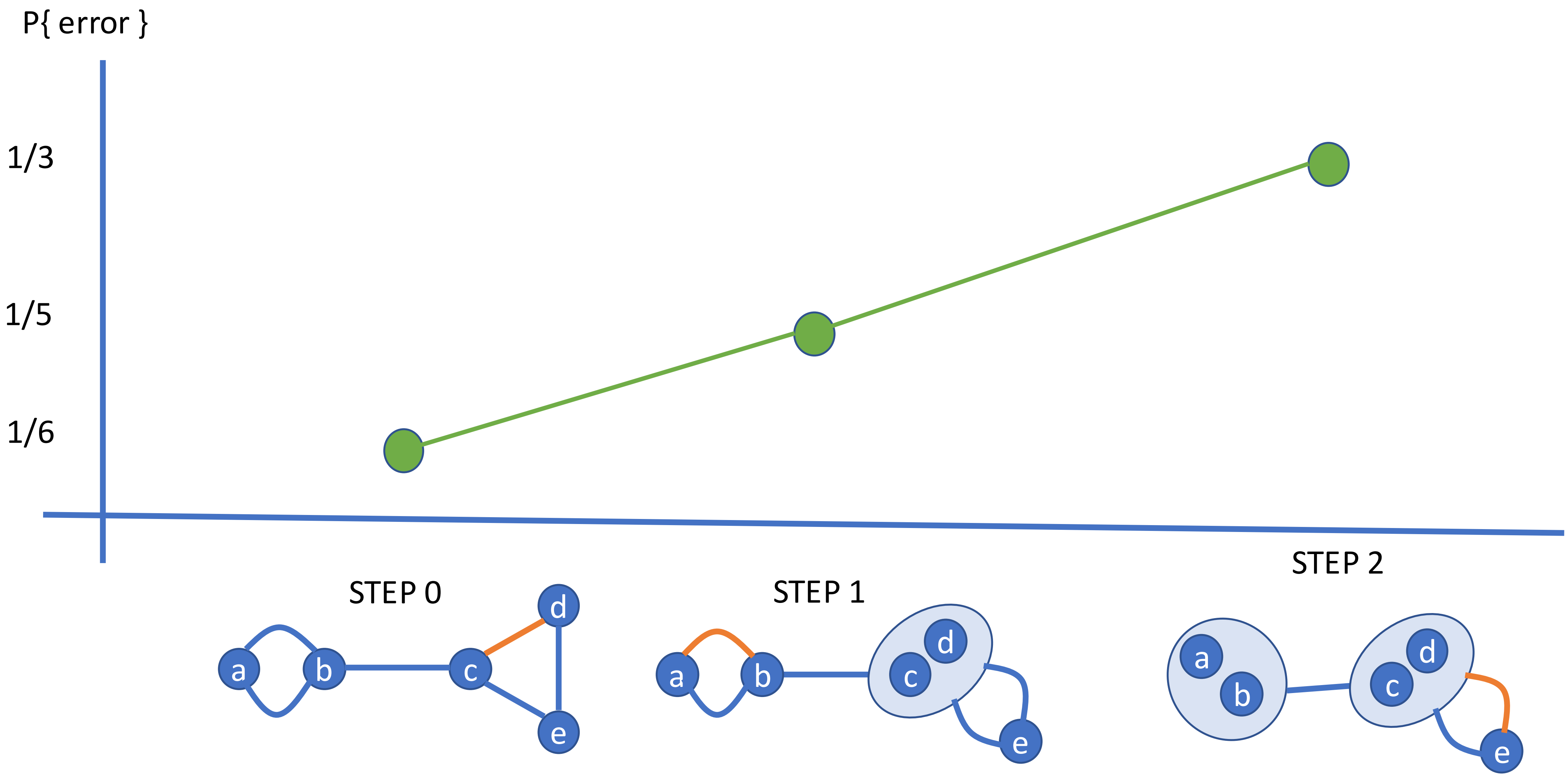
Quick Group-Work!

- Here's a successful run of Karger's algorithm.
- At each point, what is the probability of failure? (Eg, the probability that we choose an edge crossing the mincut?



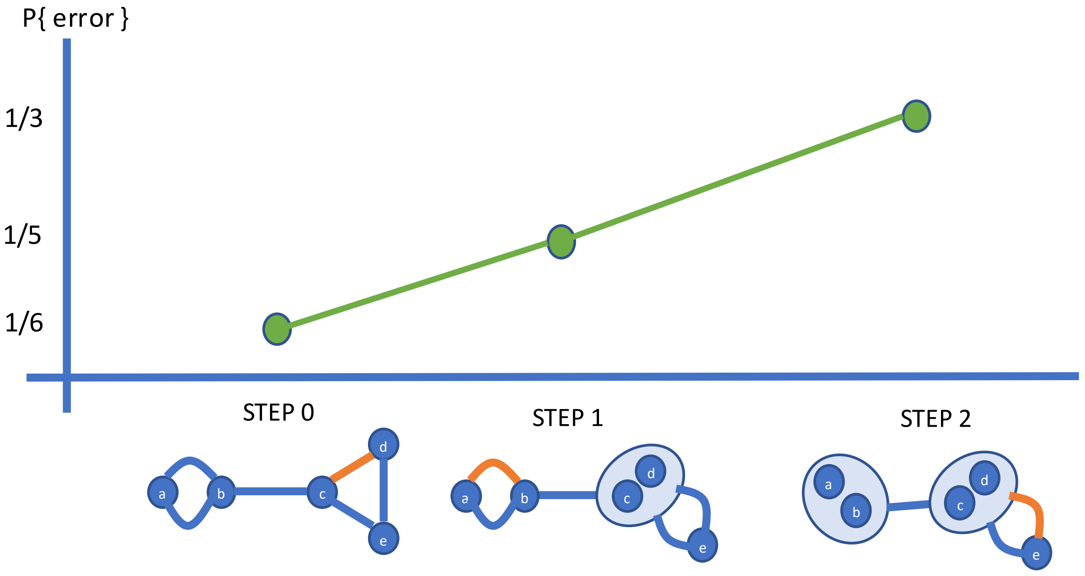
Karger's Algorithm

Any ideas for how to boost success probability better than repeating it a bunch of times?



Idea to improve

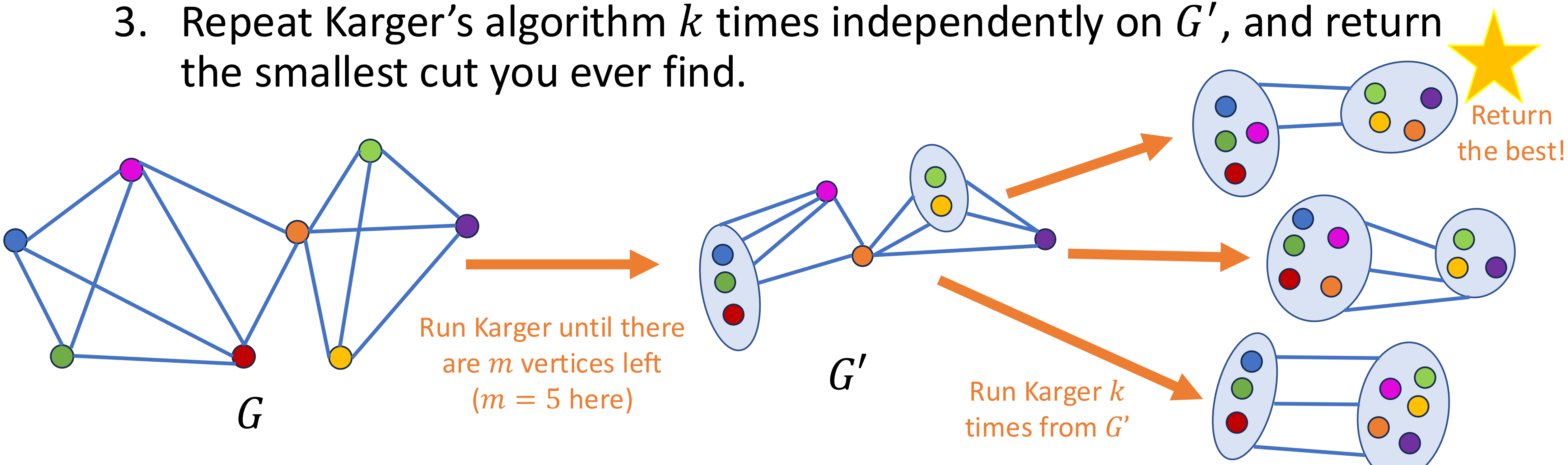
- Repeating Karger's algorithm a bunch is wasteful, because the earlier steps are much more likely to be successful than the later steps.
- Instead, repeat the later steps more than we repeat the earlier ones.



Modified-Karger

(also on the agenda document)

1. Start with a graph G on n vertices.
2. Run Karger's algorithm (once) until there are m (mega-)vertices left. Call that graph G'.
3. Repeat Karger's algorithm k times independently on G', and return the smallest cut you ever find.



Group work!

1. Start with a graph G on n vertices.
2. Run Karger's algorithm (once) until there are m (mega)vertices left. Call that graph G'.
3. Repeat Karger's algorithm k times independently on G', and return the smallest cut you ever find.

1. Give a lower bound on the probability that MODIFIED-KARGER is successful.
2. Choose $m = \sqrt{n}$ and $k = n \log n$, show that success probability of MODIFIED-KARGER is $\Omega\left(\frac{1}{n}\right)$.
3. Show that if you repeat MODIFIED-KARGER $\Theta(n)$ times, you can get a success probability of 0.999
4. Compare to Karger's algorithm from the lecture. How many edge contractions do you need?

1. Failure probability of Modified-Karger

- $P\{\text{fail}\} \leq P\{G \rightarrow G'\text{failed}\} + (P\{\text{Karger fails on } G'\})^k$

1. Failure probability of Modified-Karger

- $P\{\text{fail}\} \leq P\{G \to G' \text{failed}\} + (P\{\text{Karger fails on } G'\})^k$

$$\leq 1 - \left(\frac{n-2}{n}\right)\left(\frac{n-3}{n-1}\right)\left(\frac{n-4}{n-2}\right)\cdots\left(\frac{m+2}{m+4}\right)\left(\frac{m+1}{m+3}\right)\left(\frac{m}{m+2}\right)\left(\frac{m-1}{m+1}\right) = 1 - \frac{m(m-1)}{n(n-1)}$$

$$\leq 1 - \frac{2}{m(m-1)}$$

2. Picking parameters: $m \leftarrow \sqrt{n}$, $k \leftarrow n \log n$

- $P\{\text{fail}\} \leq P\{G \to G' \text{failed}\} + (P\{\text{Karger fails on } G'\})^k$
- $\leq 1 - \frac{m(m-1)}{n(n-1)} + \left(1 - \frac{2}{m(m-1)}\right)^k$
- $\approx 1 - \frac{1}{n} + \left(1 - \frac{2}{n}\right)^{n \log n}$
- $\leq 1 - \frac{1}{n} + e^{-2 \log n}$
- $\leq 1 - \frac{1}{2n}$ (if n is big enough)

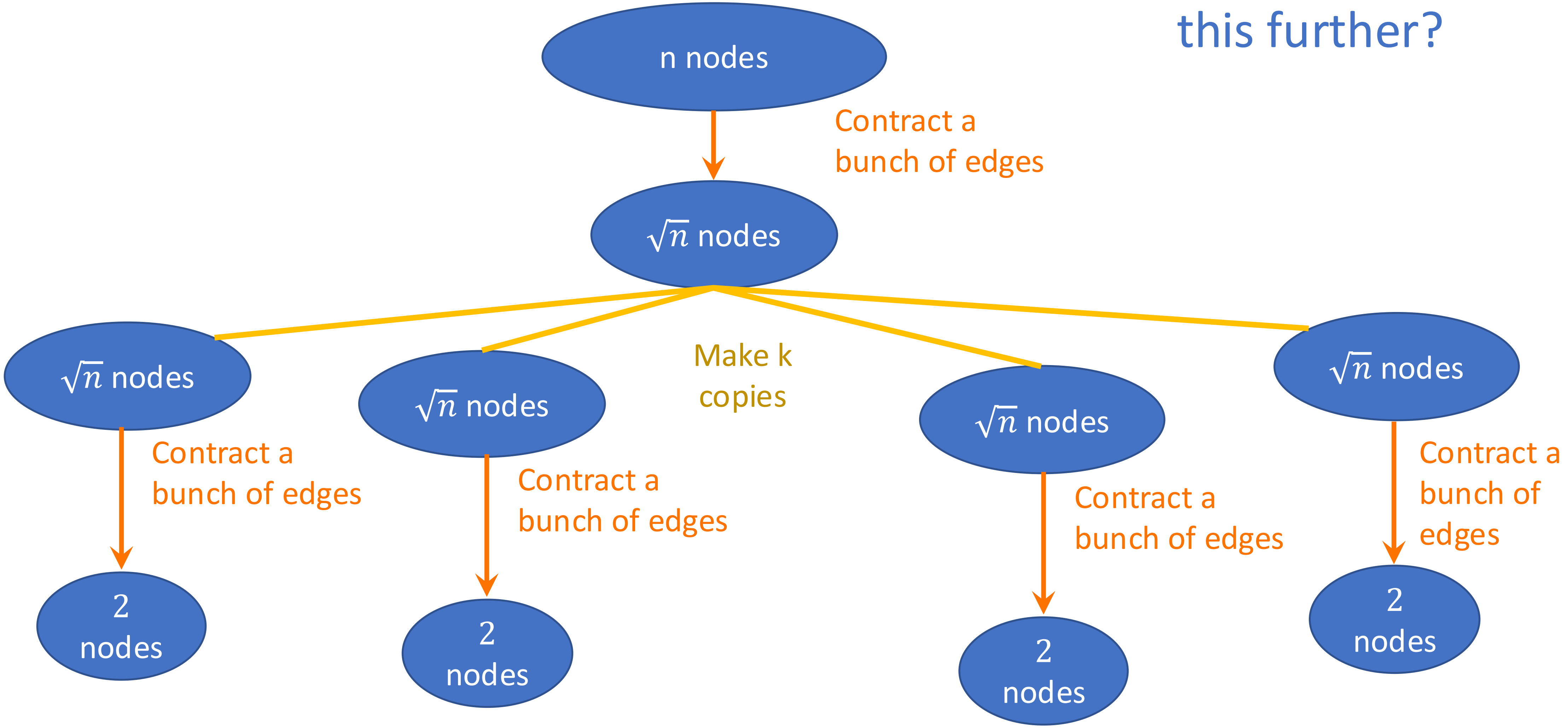
Repeating O(n) times

- 3: Failure probability?
- $\Pr\{\text{fail every time}\} \leq \left(1 - \frac{1}{2n}\right)^{100n} \leq e^{-50} \leq 0.01$

- 4: Number of contractions?
  - For one run of Modified-Karger:
    - At most n to contract down to G'
    - At most $k \cdot m = n^{\frac{3}{2}} \log n$ to contract G' k times.
    - The second part dominates.
  - Repeat the whole thing $O(n)$ times to get $O\left(n^{\frac{5}{2}} \log n\right)$.
  - Better than $O(n^3)$ contractions!

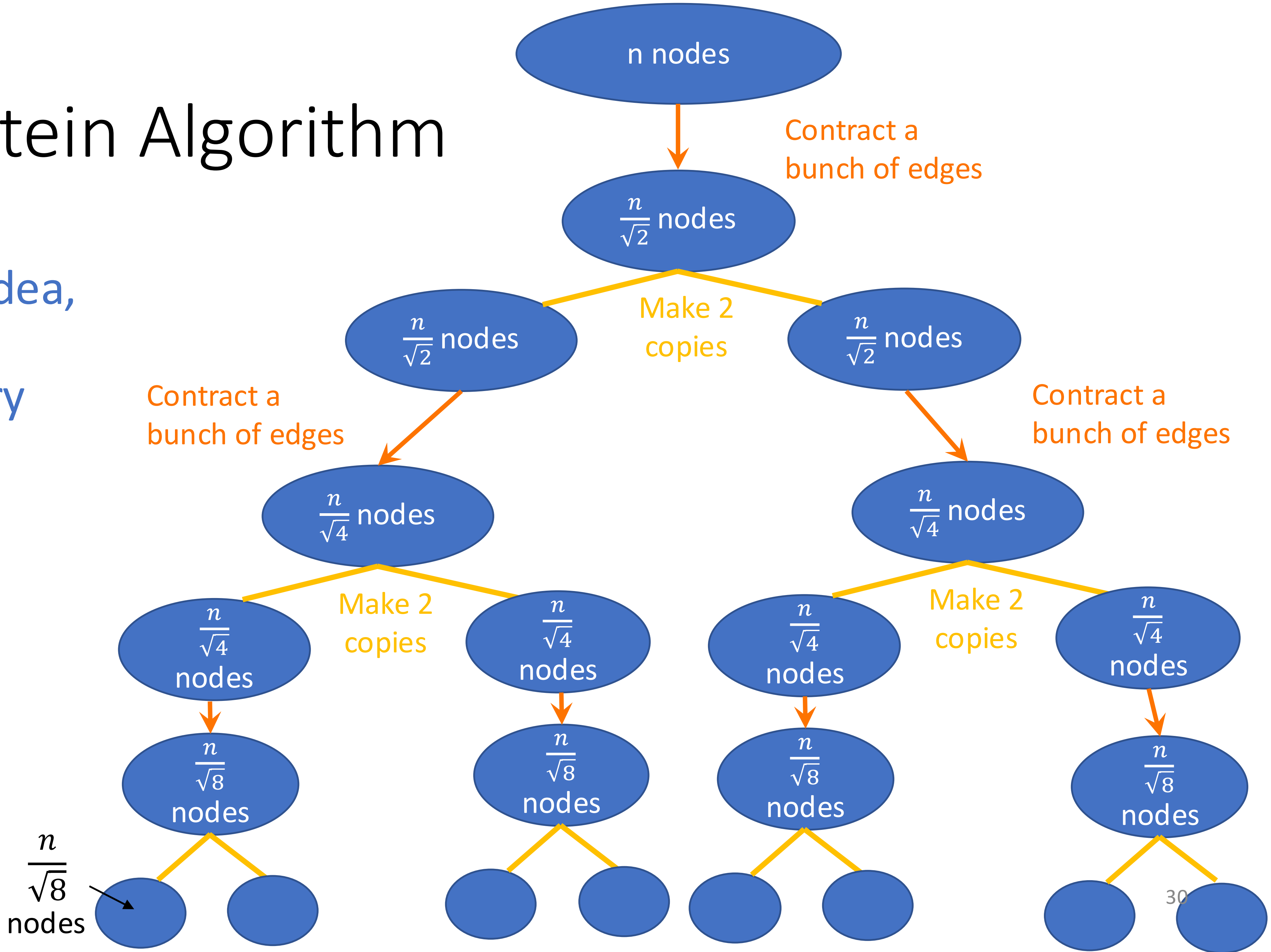
What we just did…

Suggestions for how to improve this further?



Karger-Stein Algorithm

- Same basic idea, but make a deeper binary tree!



Fun follow-ups

(for after class, if you are interested)

- Analyze this algorithm!
  - What's the success probability?
  - If we repeat it enough times to boost the success probability to 0.9999, how many contractions will we use?

- This is called the "Karger-Stein Algorithm"
  - See course website for original paper. [Karger-Stein 1996]
  - See course website for a paper with another algorithm [Karger 1998], also based on random contractions, that does even better! (Near-linear time)
  - Near-linear time deterministic algorithms? See [Kawarabayashi and Thorup, STOC 2015].

Wrap-Up

What have we learned?

- Linearity of expectation is very useful!
  - It allows us to analyze pigeon behavior
  - Coupon collecting! (More later)
  - An improvement to Karger's algorithm!

- Karger-Stein Algorithm
  - Basic idea:
    - Don't waste time by repeating earlier and later contractions the same amount
    - Later contractions are more failure-prone – repeat those more!

Before next time

- Watch videos / read lecture notes on **primality testing**!
- Do the quiz on Gradescope.