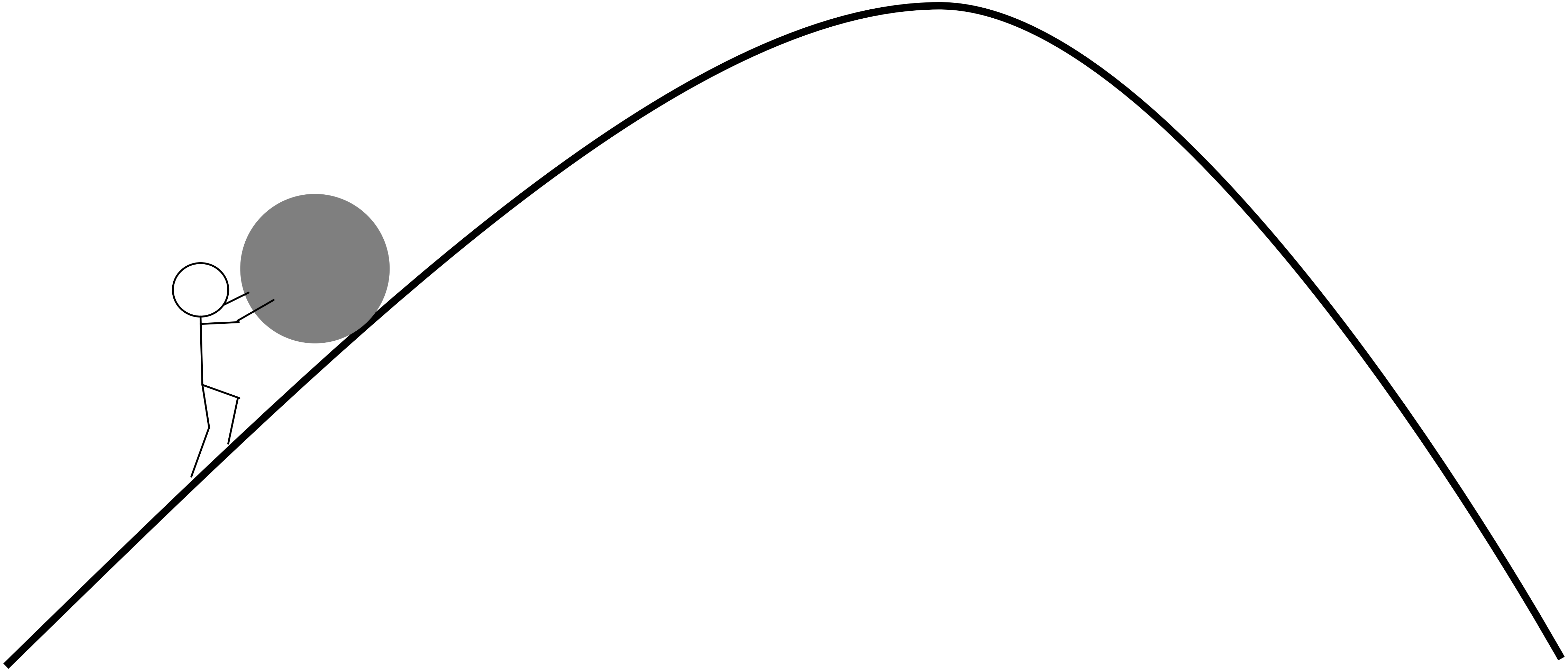
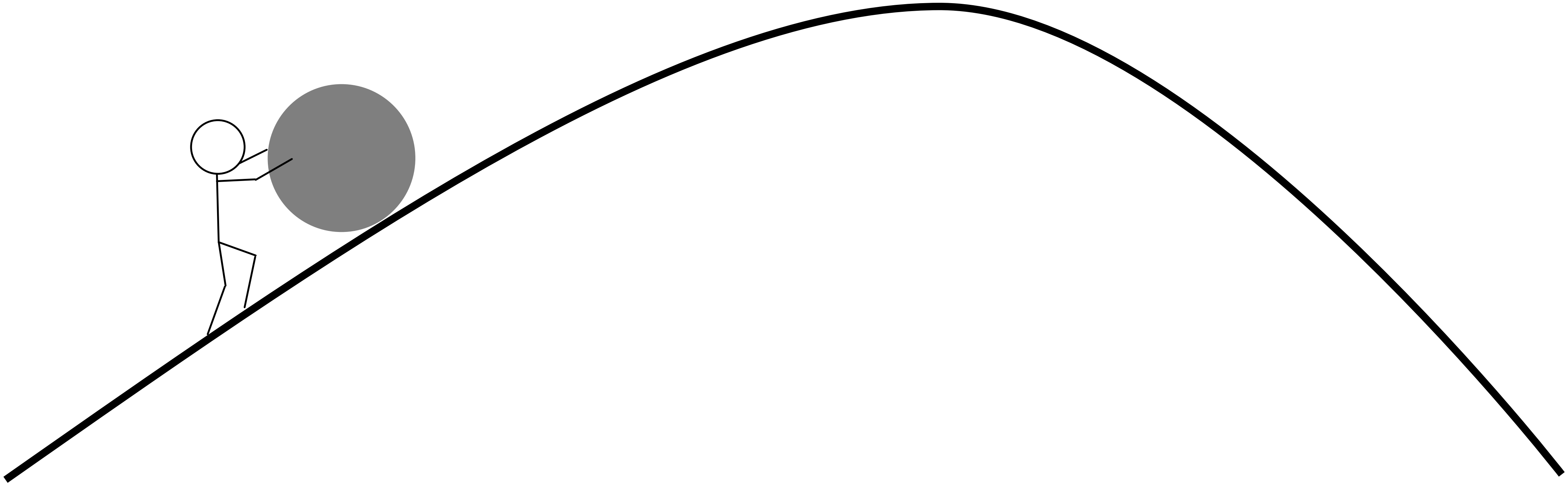
Population stability

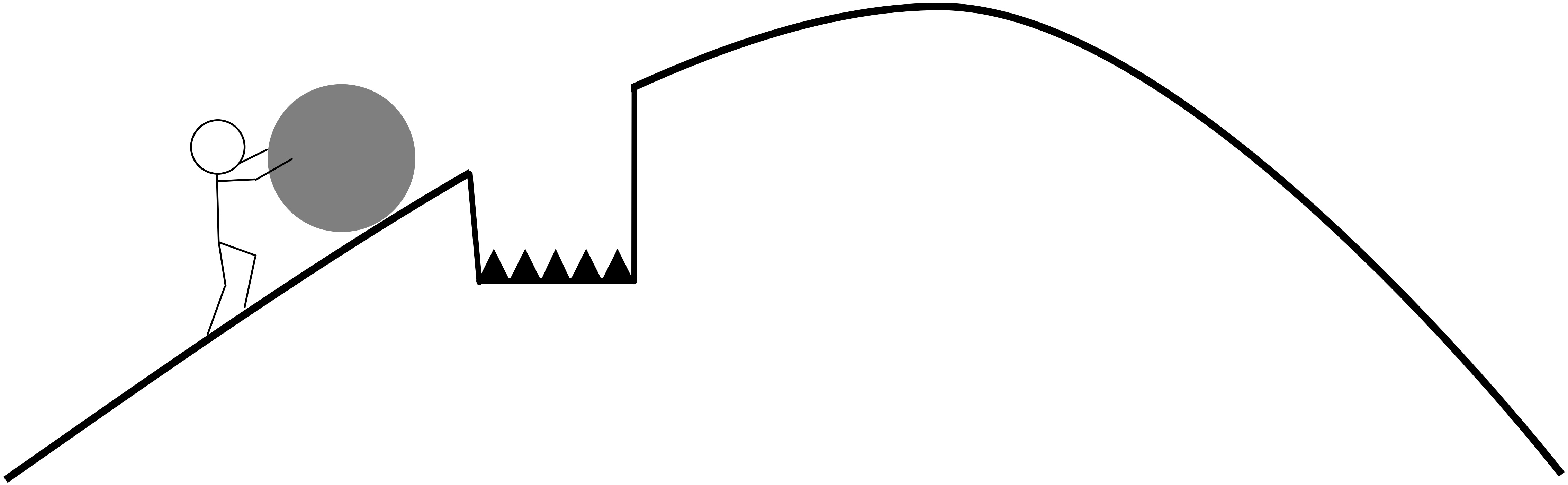
Regulating size in the presence of an adversary

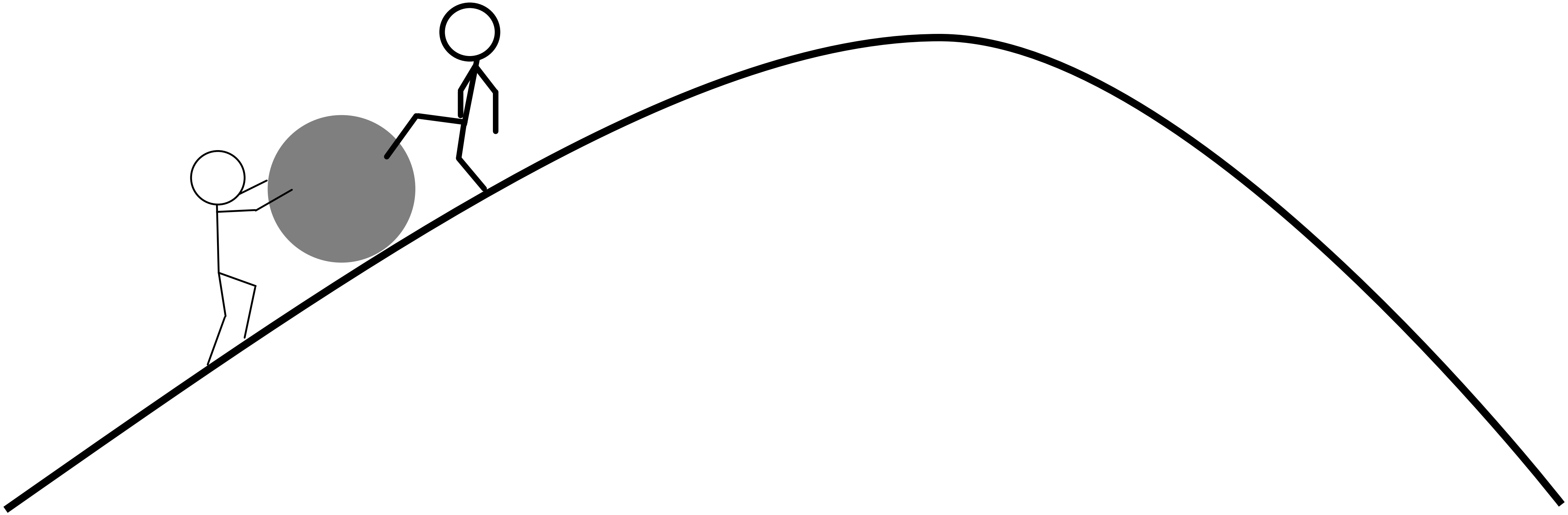
Adam Sealfon
MIT

DOE CSGF Program Review, 2018









Computing when things go wrong

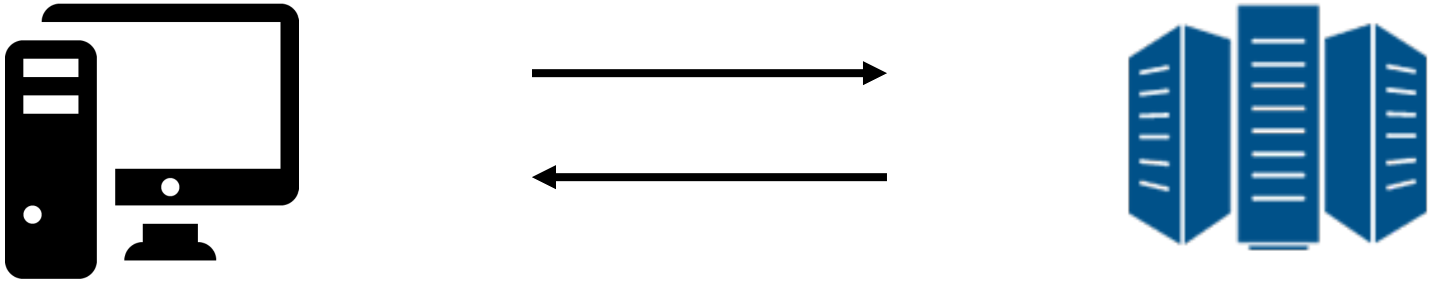
Cryptography

Privacy

Distributed algorithms

Cryptography

- How can we communicate privately so only the intended recipient can read the message?
- How can we outsource a computation to a powerful but untrusted computer, without revealing what the computation is?

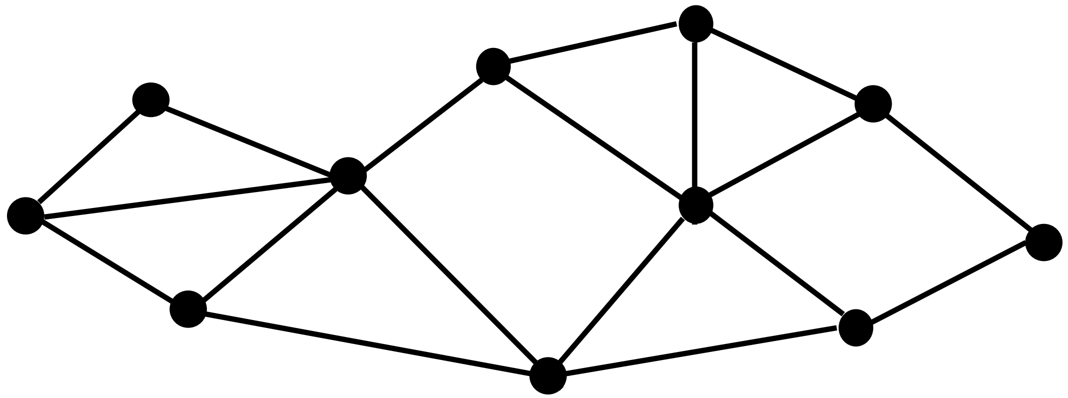


Data privacy

- How can we analyze data and publish conclusions without revealing sensitive attributes of individual data records?
- What aggregate information is safe to reveal, even approximately?

Distributed algorithms

- How can many processors coordinate to solve a problem?
- Individual processors may have limited memory, communication, processing power, and access to input.
- Processors may be unreliable, and the communication network may be prone to failures



Computing in the presence of an adversary

Cryptography

Privacy

Distributed algorithms

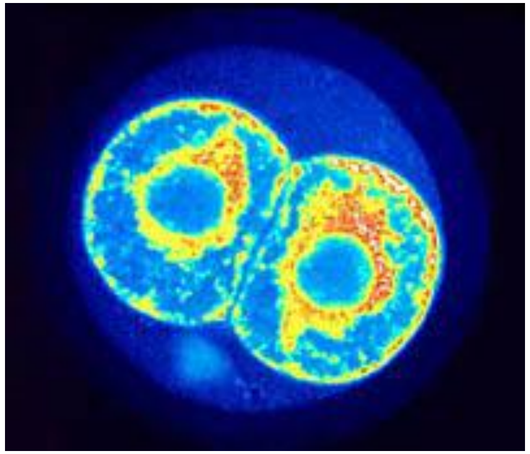
Model as a security game / experiment

Cryptography

Privacy

Distributed

Population stability



Consider a system of agents with:

- Limited memory
- Limited ability to communicate
- The ability to reproduce and self-destruct

in the presence of an adversary that can delete and insert additional agents at bounded rate.

How can we maintain a stable population size close to target value N?

Population stability—model

- Initially, there are N agents each with O(log log N) bits of memory
- Each round, pairs of agents chosen at random exchange messages and may update their state
- The adversary may insert or delete up to $N^{1/4-\varepsilon}$ agents per round

The adversary *wins* on round i if the population in that round is far from N, i.e. either > $(1+\varepsilon)$ N or < $(1-\varepsilon)$ N.

Population stability—challenges

- O(log log N) bits of memory is far too little to count the population
- No consistent communication network from round to round
- Adversary can selectively delete agents, making leader election strategies nonviable
- Adversary can insert many agents of *every possible state* in each round

Strategy: variance encoding

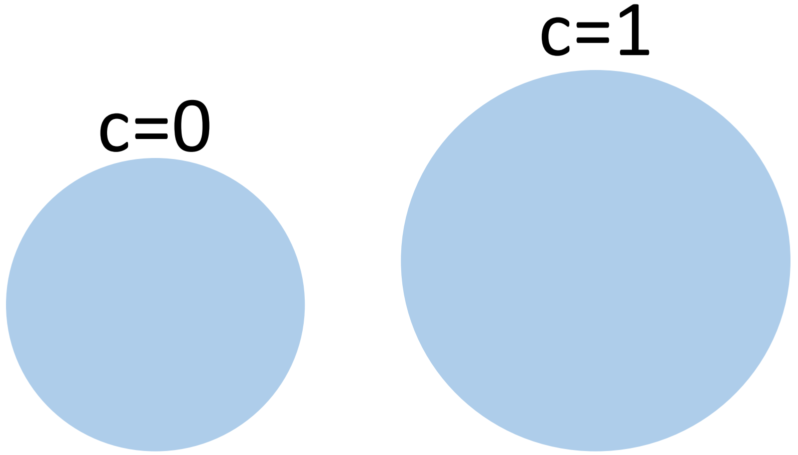
- Coordinate to sample from a distribution whose variance gives an approximation of the population size
- Each agent locally obtains a weak estimate of the variance and decides whether to replicate or self-destruct

Basic fact: if we flip a coin k times, the fraction that land heads will be roughly $\frac{1}{2} \pm \frac{1}{2\sqrt{k}}$

Local coloring

- Each agent flips a fair coin $c \leftarrow \{0,1\}$.
- Look at the colors of the next two agents.
  - Equal -> split (with probability $1 - 1/N$)
  - Unequal -> self-destruct
- The smaller the current population N’, the more imbalanced the distribution of colors, and the more likely it is that an agent will split.

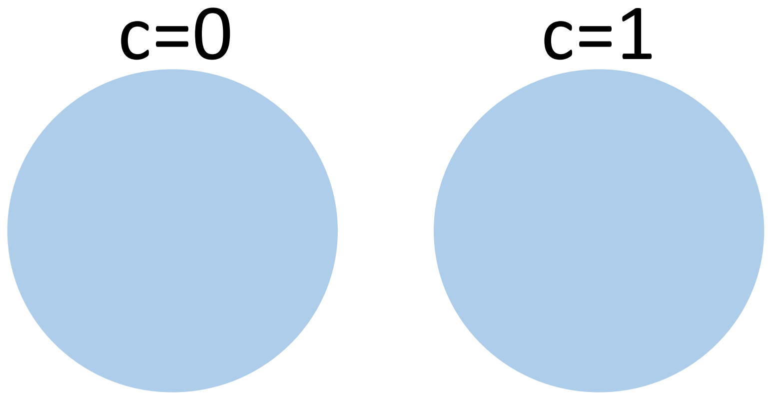
N’ small:



Local coloring

- Each agent flips a fair coin $c \leftarrow \{0,1\}$.
- Look at the colors of the next two agents.
  - Equal -> split (with probability $1 - 1/N$)
  - Unequal -> self-destruct
- The smaller the current population N’, the more imbalanced the distribution of colors, and the more likely it is that an agent will split.

N’ large:



Clustered coloring

- Only $\sqrt{N}$ agents choose random colors, and each shares its color with $\sqrt{N}$ additional agents
- Look at the colors of the next two agents.
  - Equal -> split (with probability $1 - 1/\sqrt{N}$)
  - Unequal -> self-destruct
- Strategy can be implemented in low memory
- This amplifies the signal enough to preserve the population size

Conclusion

- This protocol robustly maintains a stable population size with:
  - $O(\log \log N)$ bits of memory per agent
  - 3-bit messages
  - $O(N^{1/4-\varepsilon})$ adversarial insertions or deletions per round

- What other invariant properties can we maintain robustly?
- How can we maintain a stable population size in other models, e.g. network-based communication?
- Can we use these ideas to obtain more robust protocols for approximate counting?

Acknowledgements

- Shafi Goldwasser
- Rafail Ostrovsky and Alessandra Scafuro
- Aydın Buluç and Ariful Azad
- DOE CSGF and Krell