GATEway

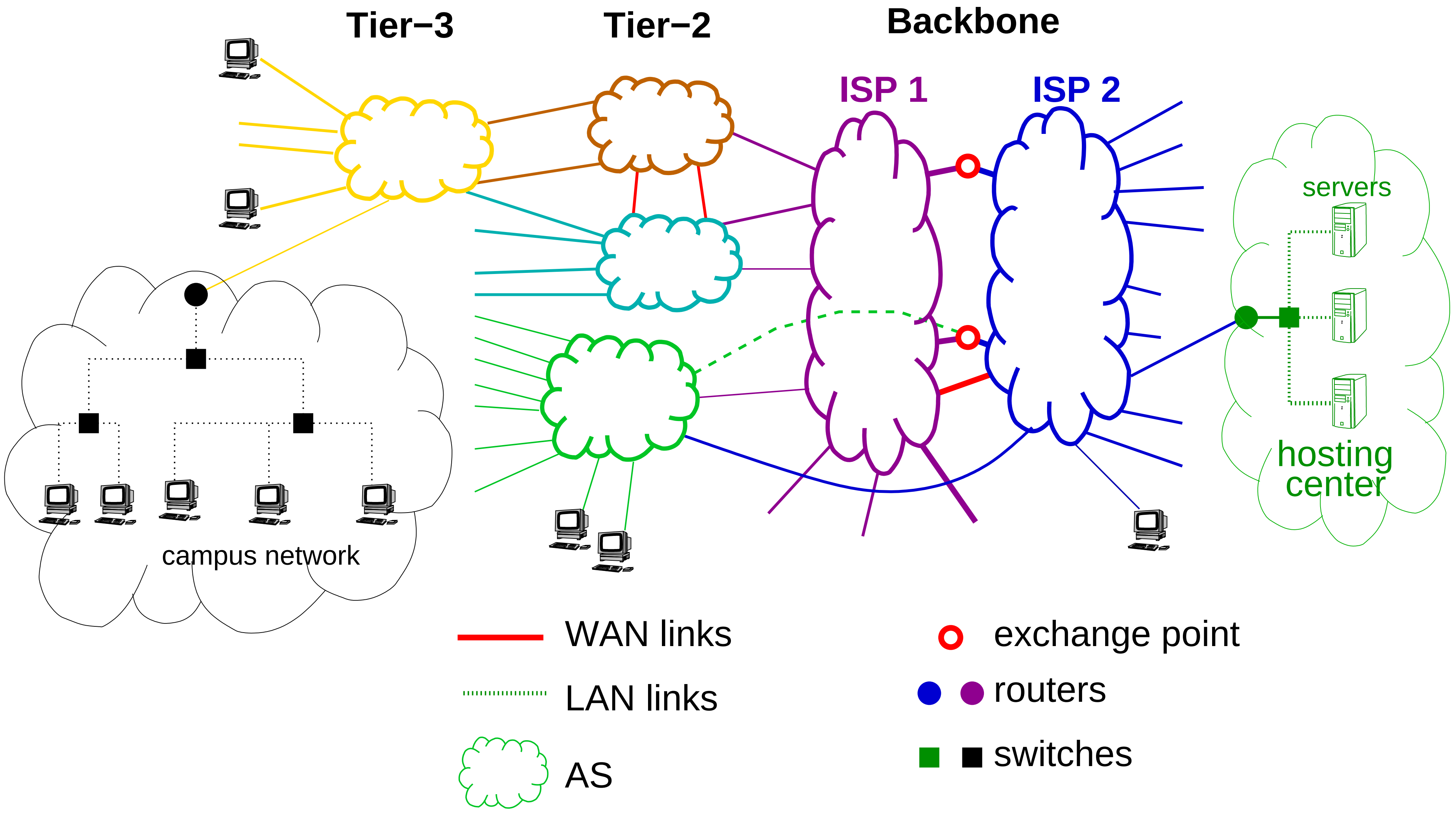
Symbiotic Inter-Domain Traffic Engineering

Matthew Roughan and Yin Zhang

<matthew.roughan@adelaide.edu.au>

Discipline of Applied Mathematics
School of Mathematical Sciences
University of Adelaide

A Cartoon of the Internet



The Problem

- The Internet is a "Network of Networks"
  - Autonomous Systems (ASs) $>$ 20,000 of them
  - There is a vast range of types of AS
- They are independently managed
  - No central authority
  - Operators can choose their own
    - topology (network design)
    - technology
    - routing protocols and policies
  - More like a federation of networks
- These networks are competitors
- **But they must co-operate**

The Prisoner's Dilemma

		Prisoner B	
		stays mum	squeals
Prisoner A	stays mum	6 months each	B: goes free A: 10 years
	squeals	A: goes free B: 10 years	Each serves 5 years

- Prisoner's are both better off if they co-operate
- Acting individually they are better off squealing
- Critical issue is **trust**

Network operators often find themselves in a similar situation (only better because no-one goes to jail).

Traffic Engineering

- Traffic engineering means optimizing the flow of traffic
  - often called simply “Load Balancing”
- Better distribution of traffic
  - network more efficient
  - can improve performance by alleviating congestion
- Many optimization approaches to solve different versions of this problem depending on
  - objective
  - available technology
  - other constraints

Shortest-path optimization

Standard intra-domain TE problem

- Objective: Minimize maximum load on links
- Technology: Shortest-path routing
  - many networks use shortest-path routing
    - internally
    - e.g. OSPF, IS-IS
  - "shortest" but link "distances" are arbitrary
  - routers balance load across equal-cost paths
    - but this isn't critical to get good results
- standard approaches to optimization
  - aim to chose link distances such that routing balances traffic

Solutions

- Shortest-Path Optimization Problem is NP-hard
  - need heuristic solution technique
  - there are several available
- We rolled our own
  - to make it work easily in what follows
- Genetic Algorithm
  - genes store link “distances”

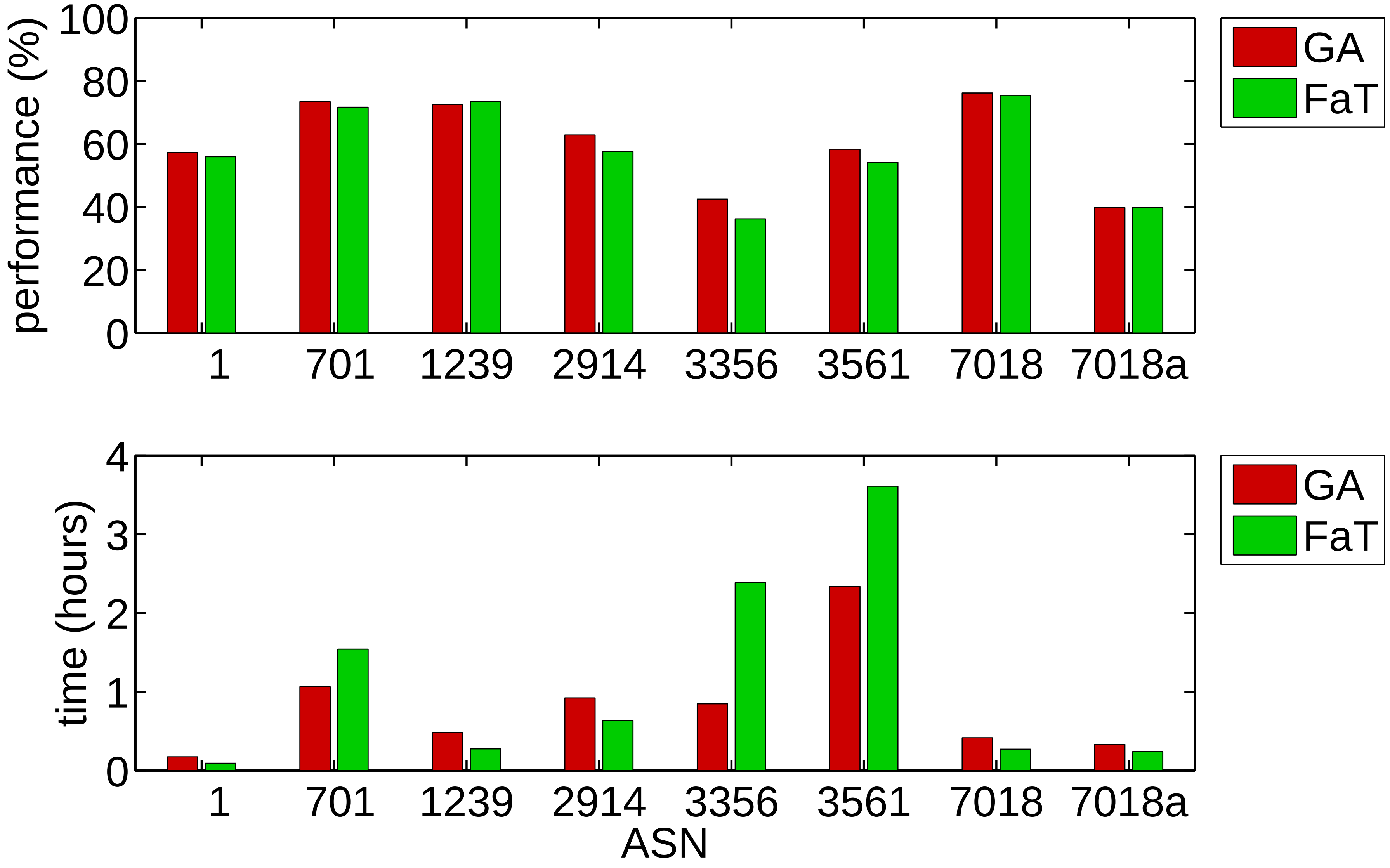
Simulations

- Traffic from a gravity model
- Networks from Rocketfuel

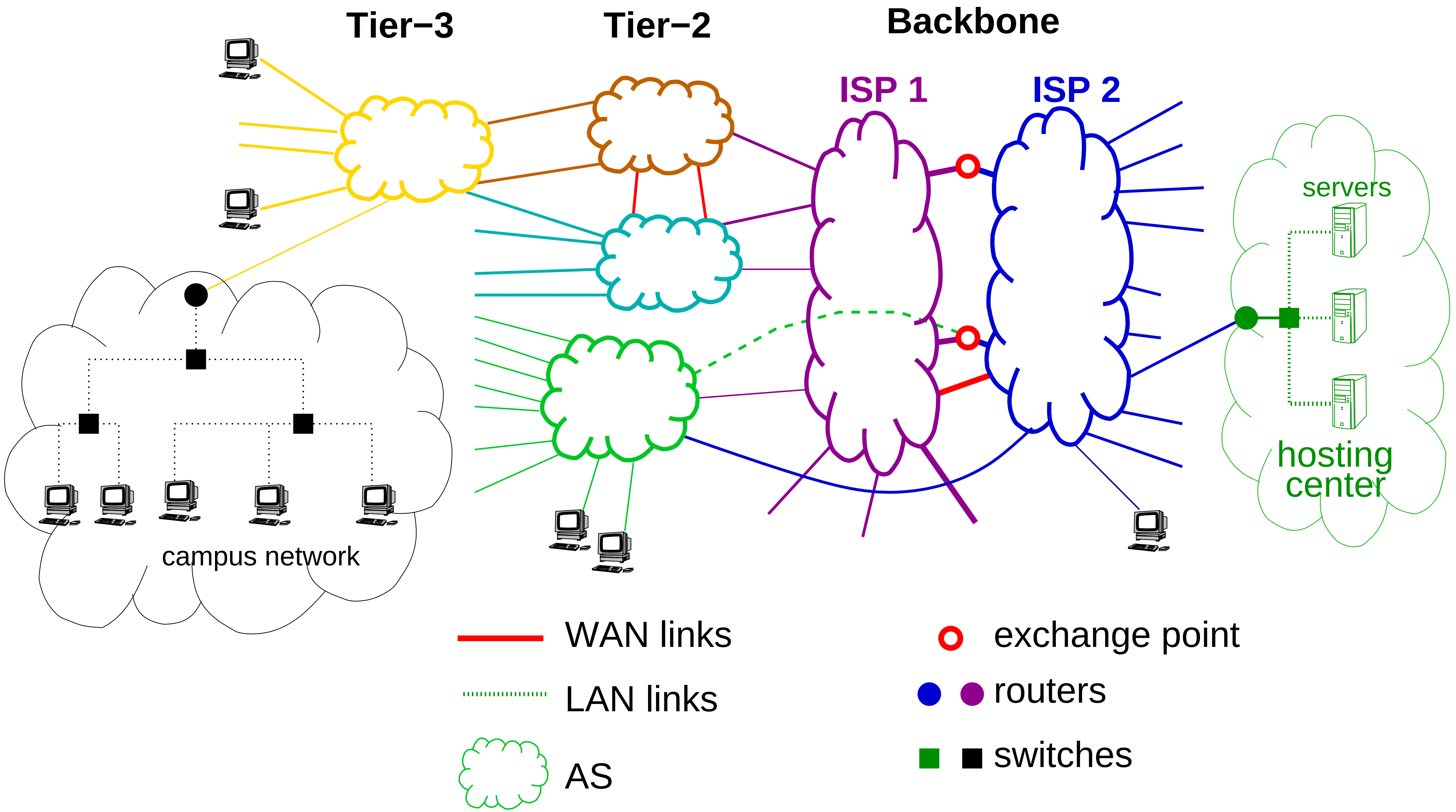
ASN	Name	Nodes	Links
1	Genuity	24	74
701	UUNet	48	368
1239	Sprint	33	130
2914	Verio	47	176
3356	Level 3	46	536
3561	Cable & Wireless	59	592
7018	AT&T	35	136

based on measurements of real networks.

GA Opt. on a single network



Inter-domain TE



TE is a bit more complicated between domains

2 Networks

Look at the simple problem of two networks

- Networks join at multiple points
  - balance loads internally, and across joins
- Routing is no longer shortest-path (BGP)
- The objective can be the same
- Co-operation is necessary to allow optimization
- Joint optimization requires revelation of information that competitors would rather keep secret!
  - network topology and routing
  - traffic loads

Selfish behaviour

- Typical providers behave selfishly
  - won't reveal information
  - separately optimize their own networks
  - e.g. hot potato routing
- The result is clearly worse than if they co-operate, but they need to establish a way of co-operating while maintaining secrecy of their private information
- There seems to be no solution

Another similar problem

Millionaires' problem

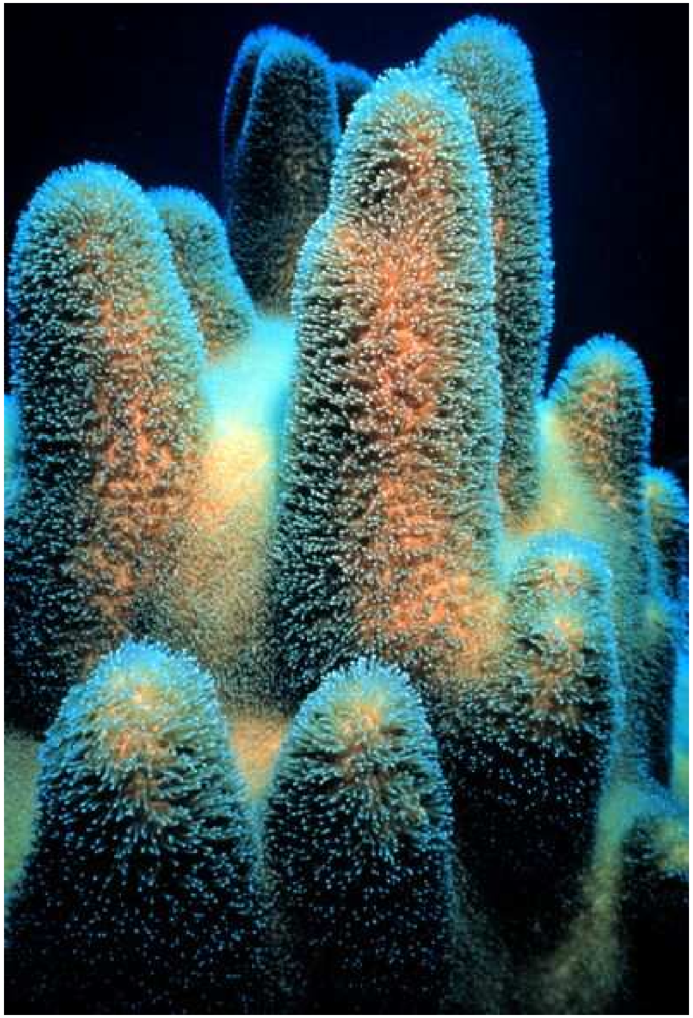
- Bill Gates and Warren Buffet are trying to decide who should put more money into the Gates foundation $(*)$
  - they want to know who is richer
- But they are feeling rather secretive, and don't want to reveal their true wealth.
- how can they decide?

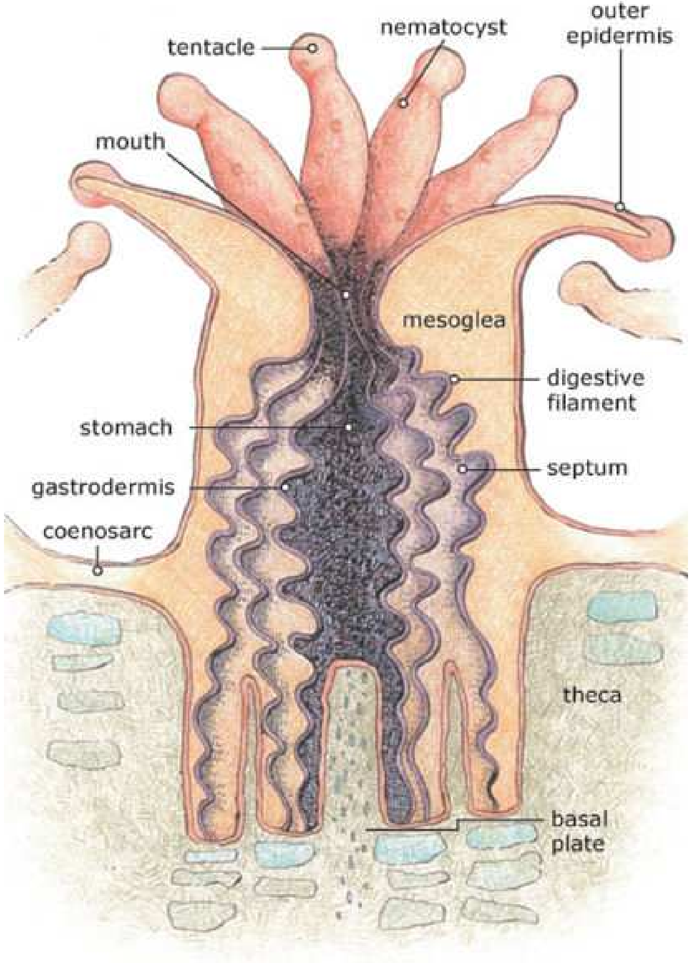
$(*)$ – no real millionaires were harmed in the production of these slides

Secure-Distributed Computing

- general solutions to such problems exist
  - secure-distributed computing
  - privacy-preserving data-mining
- Yao developed a (2 party) protocol to solve all such polynomial time problems without revealing inputs
  - typically based on cryptomaths
  - lots of extensions exist
- problem is making them efficient enough for real applications
  - our problem isn't even polynomial time
  - we have a different approach

Symbiosis





Symbiosis is a nice metaphor for privacy preservation

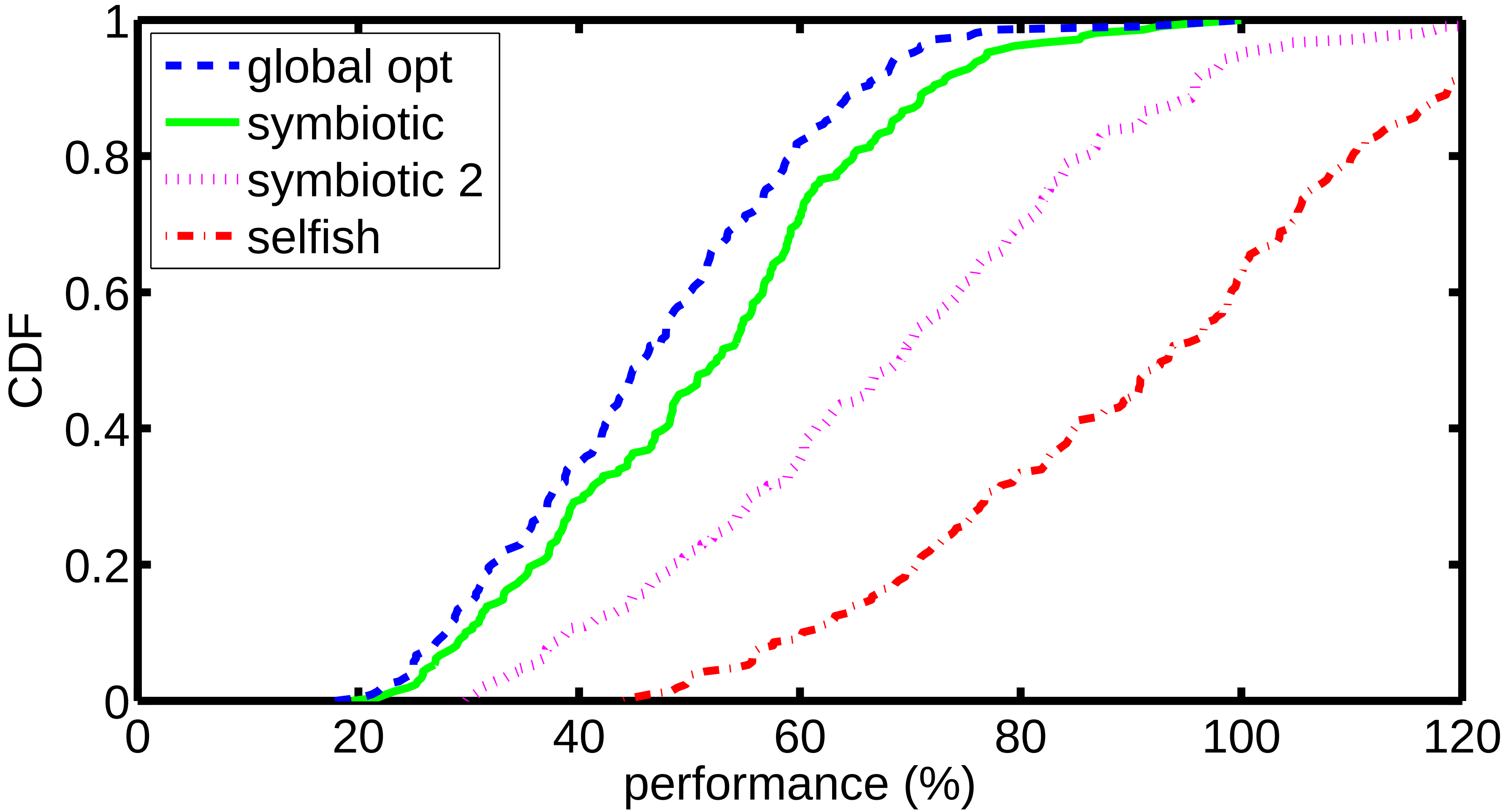
GA for Symbiosis

Given GA as a metaphor, let's include symbiosis

- each network keeps some of the "genes"
  - its own link distances
  - keep genes private
- share only information needed for fitness evaluation
  - most of this is publicly measurable anyway
  - allows joint selection
- use of same random seeds
  - allows same random selection and mutation decisions for both

Performance

Random pairs of networks

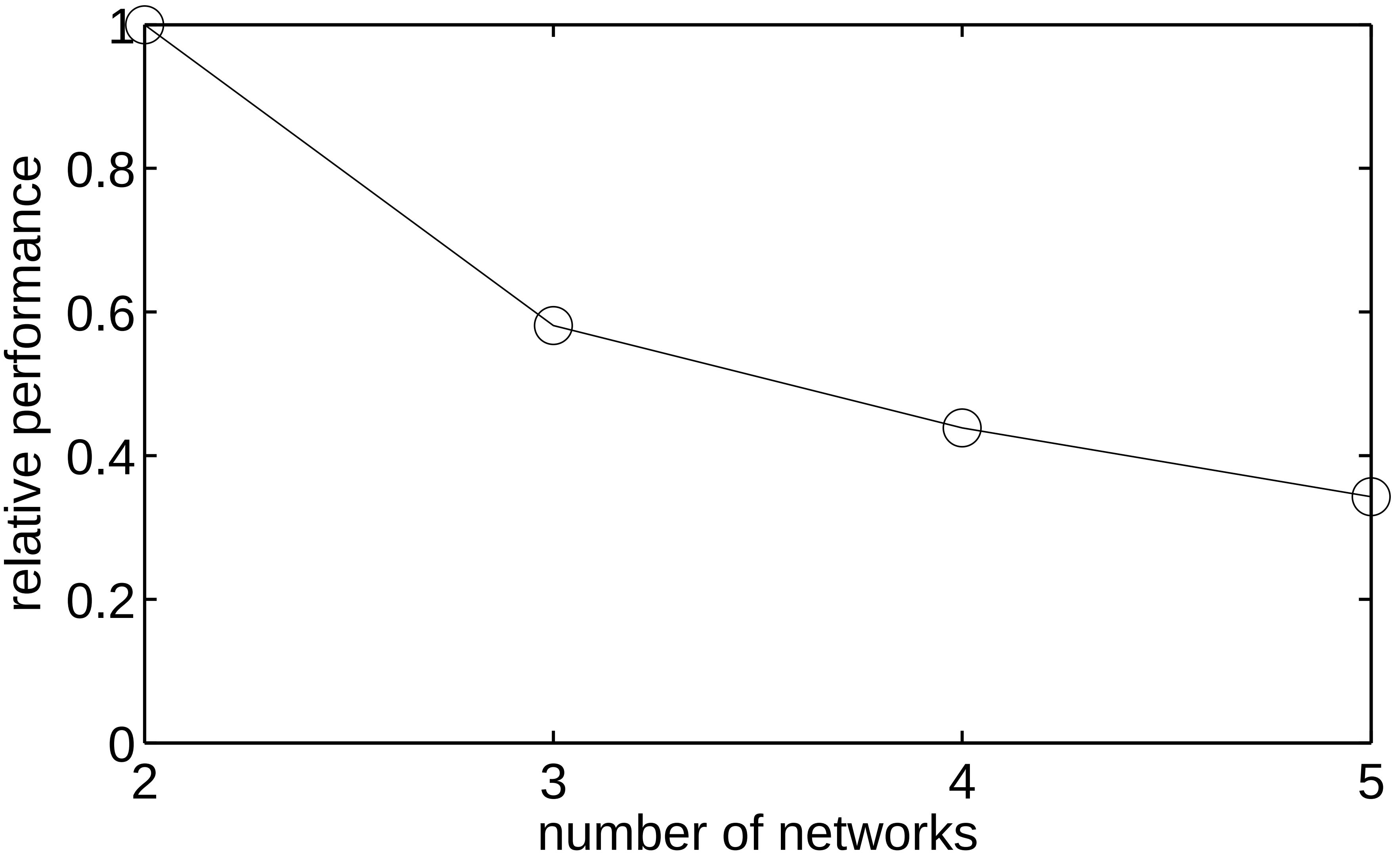


measured relative to unoptimized network

Other features

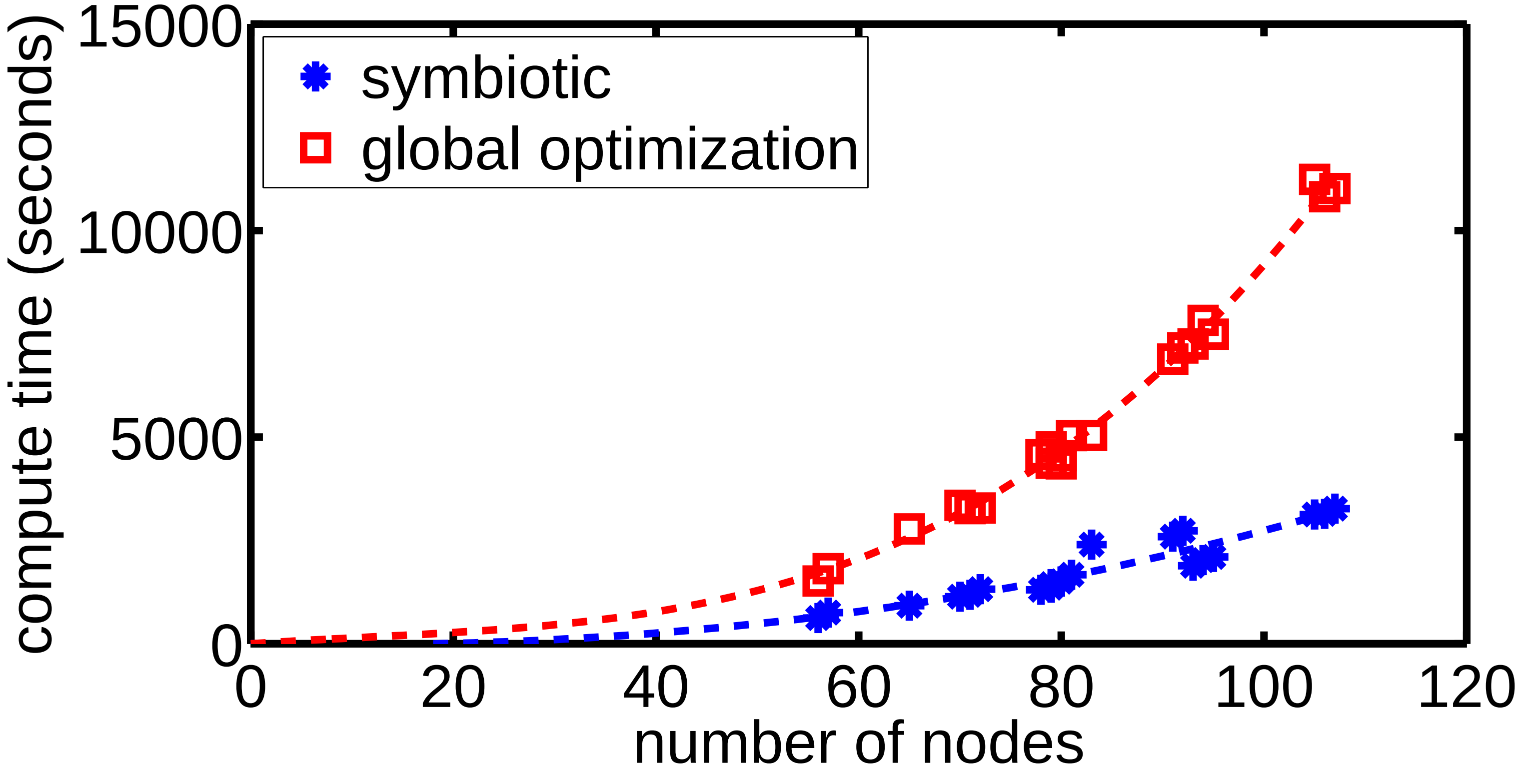
- flexible
  - alternative objective functions
  - additional constraints
  - tradeoff information leakage vs performance
    - symbiotic 2
- approximately better for larger networks
- robust to errors in inputs
- better performance for more networks

Better with more networks



Cost for privacy

- some communications overhead (not huge)
- computation times actually better!
  - $O((N_1 + N_2)^3)$ vs $O(N_1^3 + N_2^3)$
  - and calculations are distributed



Privacy-max

- second approach reduces information leakage, but reduces performance
- can maintain performance, but cost is communications overhead

Approach	Comm.s cost	Av. Perf.
joint SP	$O(N^2 + EK)$	46.6%
symbiotic	$O(GPN_{\max} \log Q)$	51.5%
symbiotic 2	$O(GPN_{\max} \log Q)$	68.4%
privacy-max	$O(GPE^2N^2)$	51.5%
selfish	zero	91.2%

Notation

G is the number of generations of the GA
P is the population size in the GA
N is the total number nodes ($N_{\max}$ is the maximum of N_1 and N_2 the number of nodes in each network).
E is the number of edges
Q is the number of inter-AS edges
Weight range $[0, 2^K - 1]$

Conclusion and Future Work

- We have an approach that can allow co-operation (to optimize load balancing)
  - preserves privacy of majority of secret information
  - analogous to symbiosis in biology
  - has some good properties
    - flexibility
    - robustness
- future
  - used semi-honest model here – apply to more general antagonist
  - apply to more general optimization problems

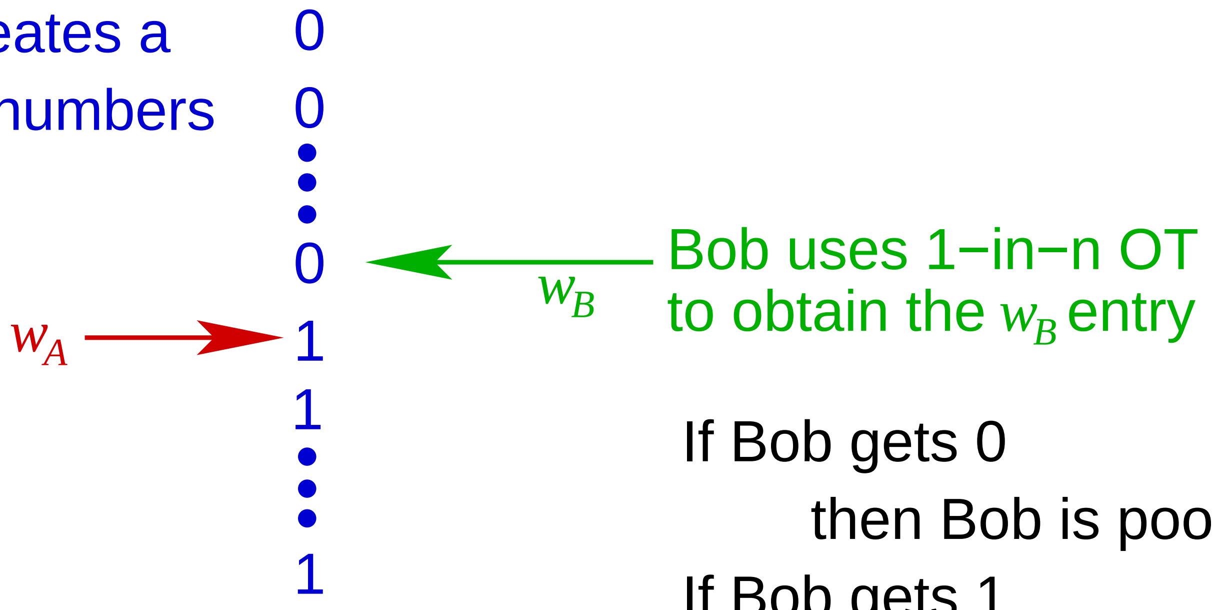
Oblivious transfer

- there are various versions
- consider 1-in-n Oblivious Transfer (OT)
  - Alice has a list of numbers $\{a_1, a_2, \ldots, a_n\}$
  - Bob has an index β
  - Bob wants to learn a_β
  - Alice must not learn β, and Bob must not learn a_i for any $i \neq \beta$.
- Bob learns exactly one item from Alice's list, without Alice learning which item Bob discovered.

Applications

- the millionaires problem
  - more generically: calculating a minimum
- Assume Alice has wealth $w_A \in [1, n]$, and Bob has $w_B \in [1, n]$, where n is known to both

Alice creates a list of n numbers



If Bob gets 0
then Bob is poorer
If Bob gets 1
then Bob is at least as rich