From Queries to Bandits:
Learning by Interacting

Lev Reyzin
Algorithms and Randomness Center
Georgia Institute of Technology

Interactive Learning

Supervised Learning

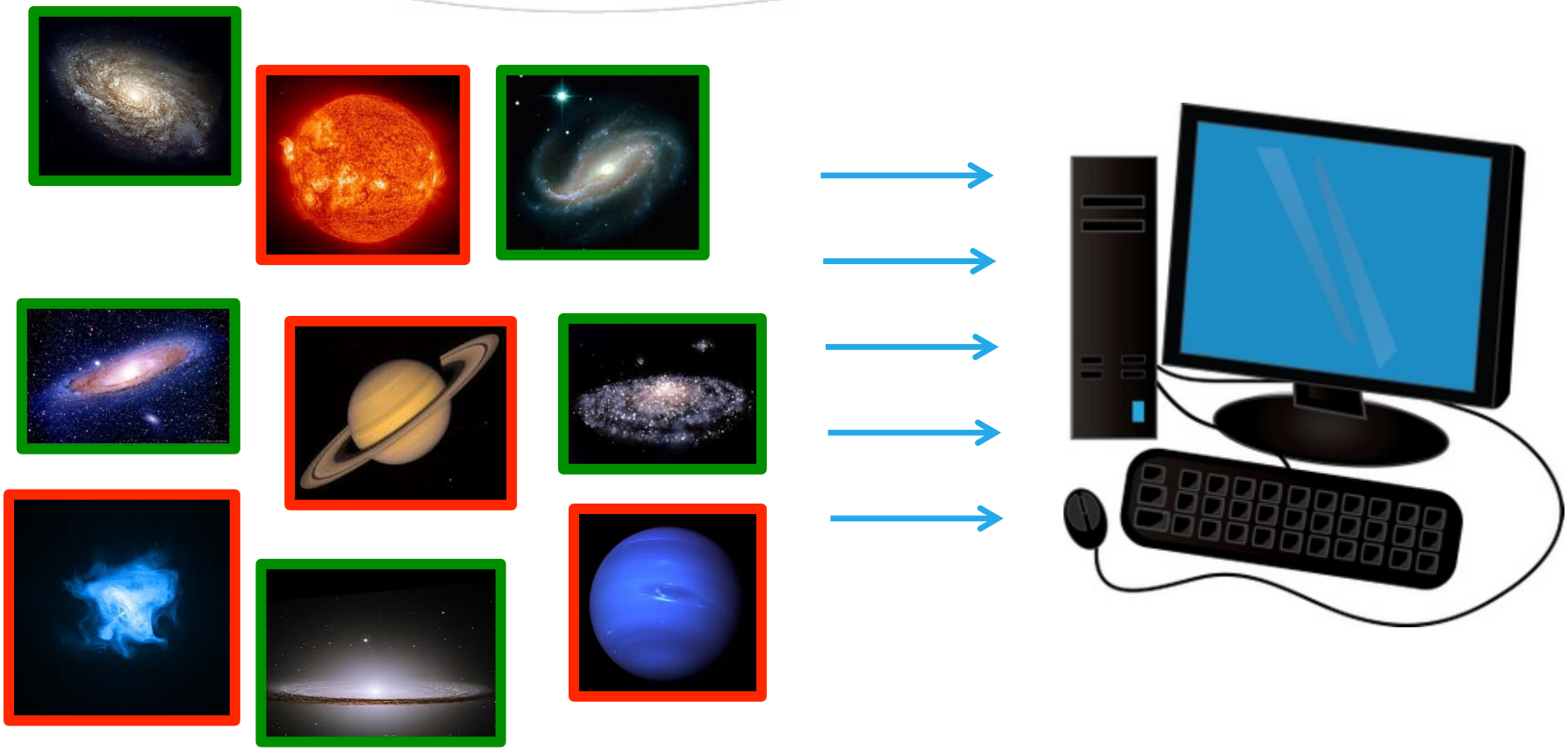
PAC Learning [Valiant '84]

Say we want a computer to learn to recognize galaxies.

A computer is first trained on labeled data.

Supervised Learning

PAC Learning [Valiant '84]

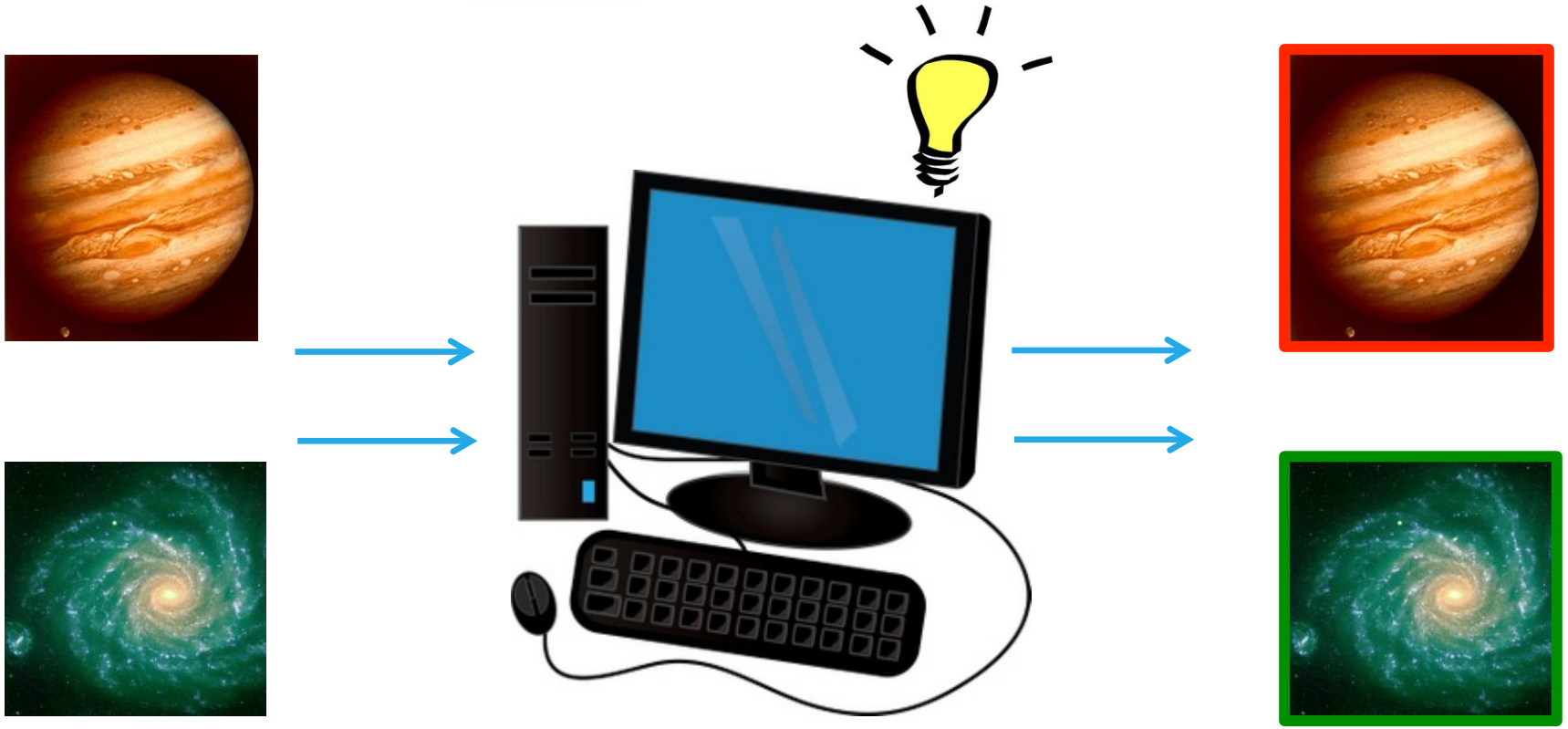


Supervised Learning

PAC Learning [Valiant '84]

Supervised Learning

PAC Learning [Valiant '84]



Problem

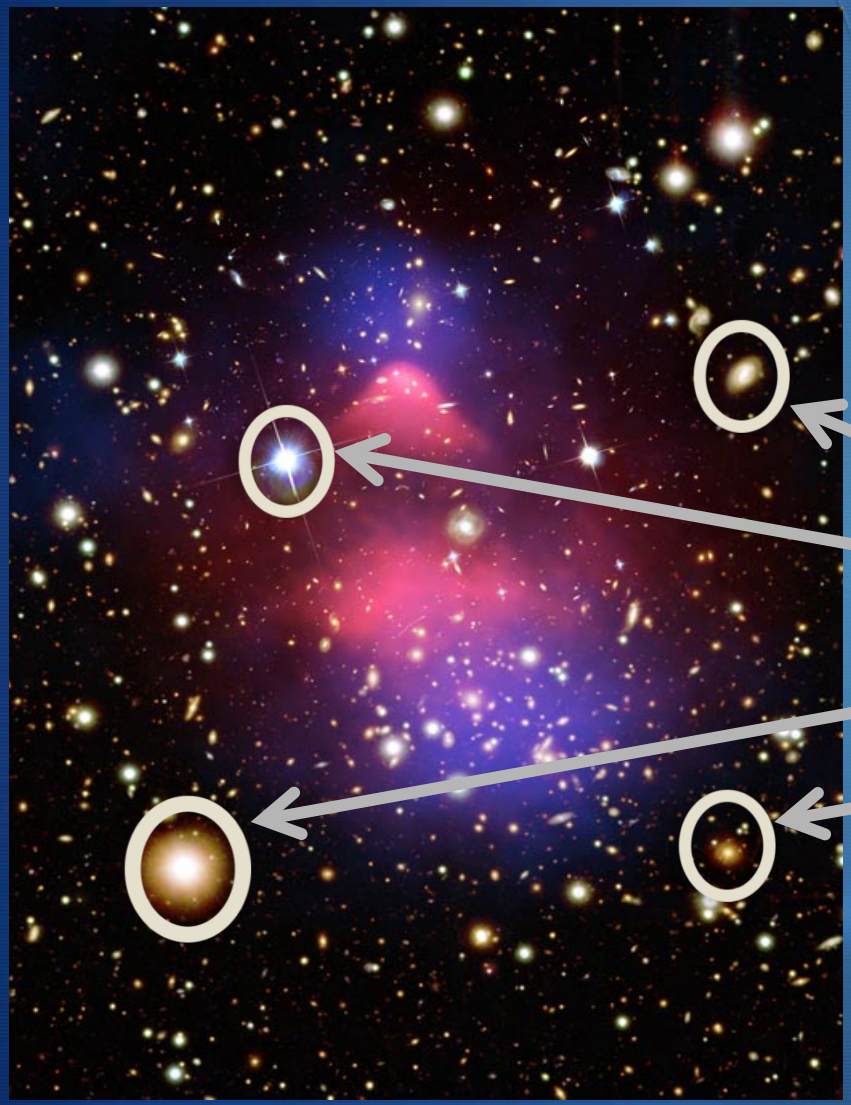
Producing training data is expensive!

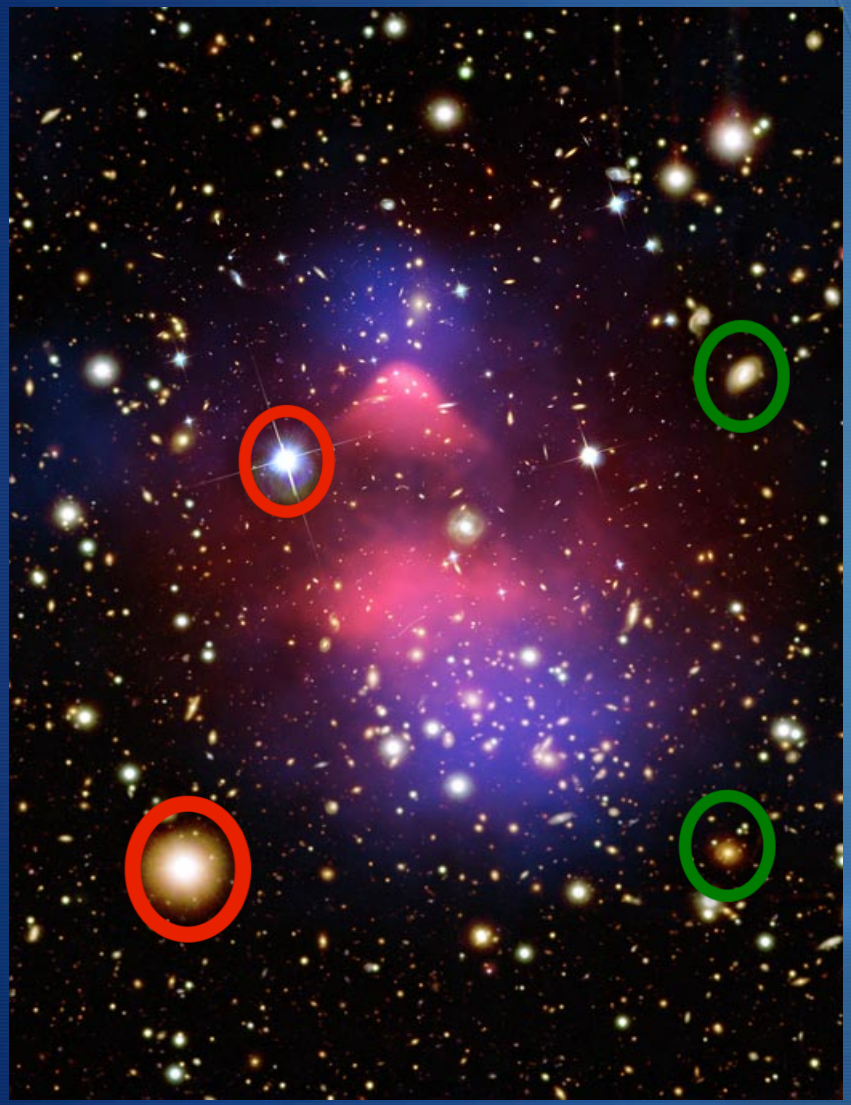
Active Learning

[Cohn et al. ’94]
[Lewis and Gale ’94]

Active Learning

[Cohn et al. ’94]
[Lewis and Gale ’94]



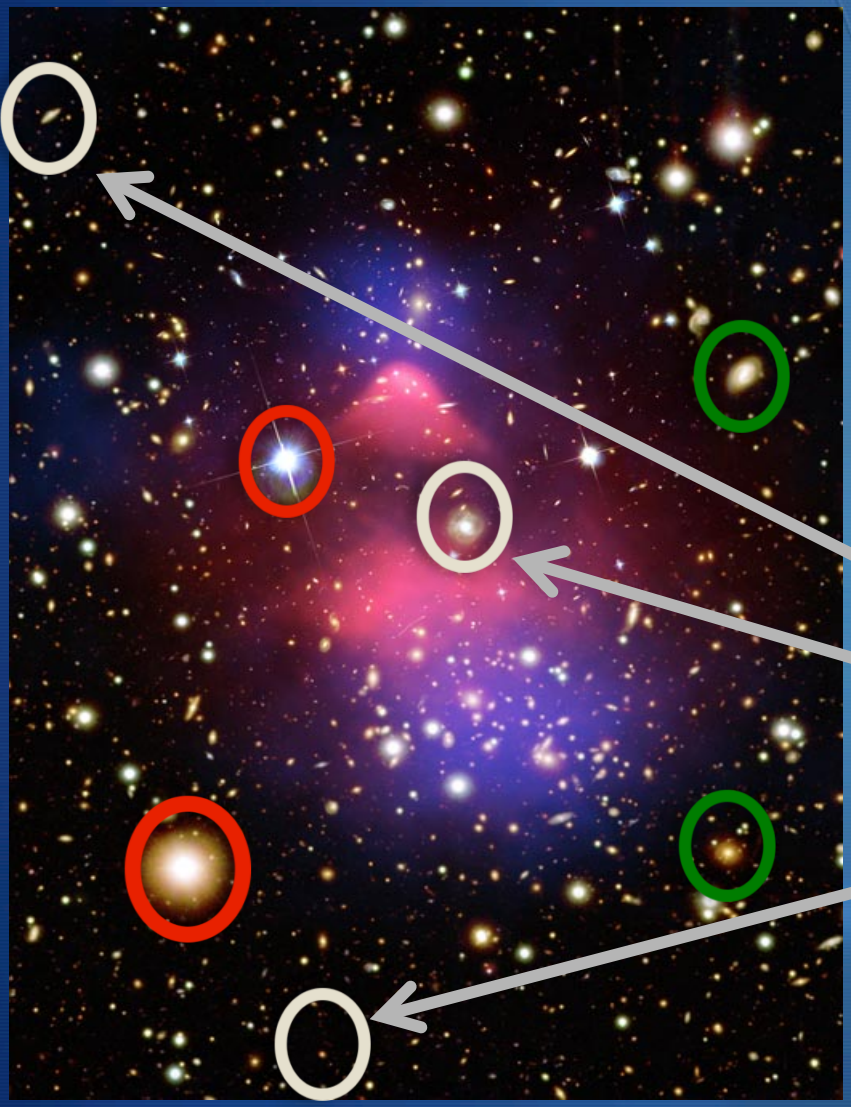


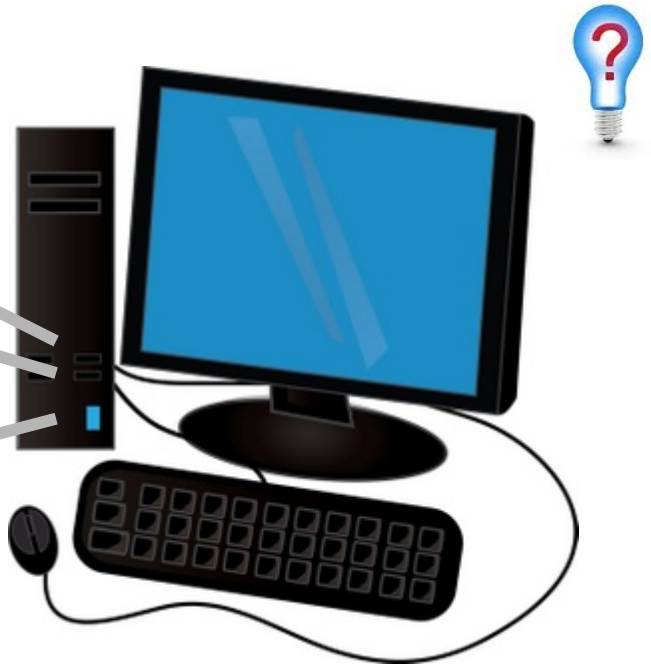
Active Learning

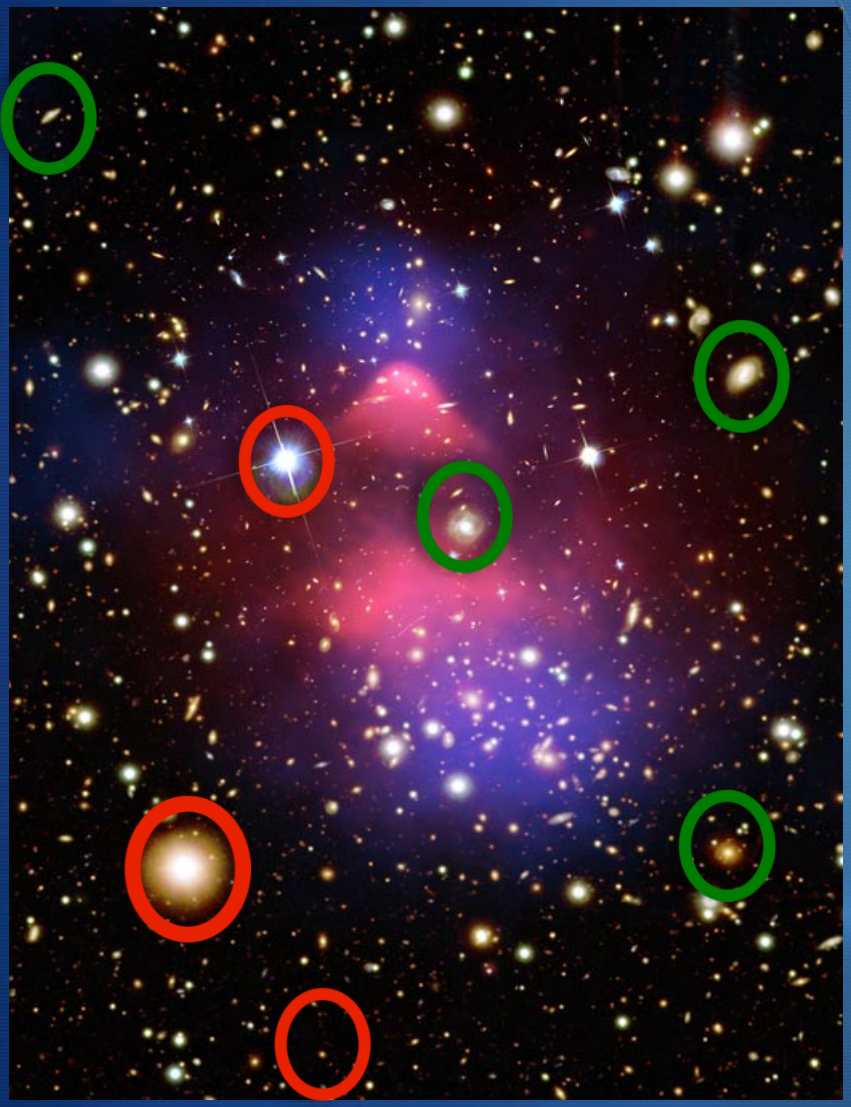
[Cohn et al. ’94]
[Lewis and Gale ’94]

Active Learning

[Cohn et al. '94]
[Lewis and Gale '94]



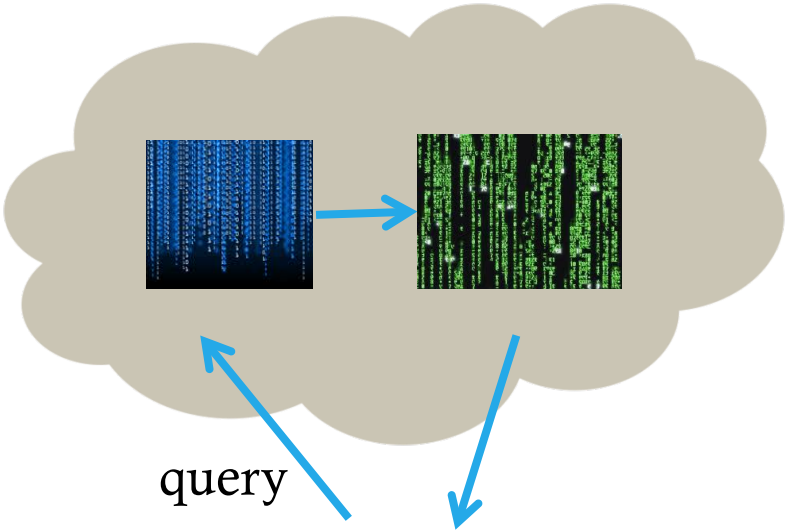




Active Learning

[Cohn et al. ’94]
[Lewis and Gale ’94]

Passive vs. Interactive



Active Learning of Interaction Networks

subject of my Ph.D. dissertation

Interaction Networks

An interaction network is a finite population of elements whose state may change as a result of interacting with other elements according to specific rules.

Interaction Networks

e.g.

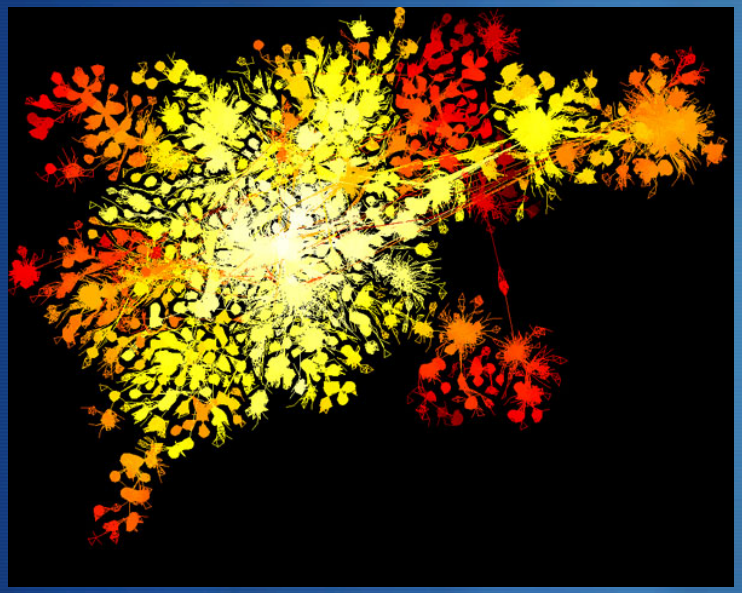
circuit

Interaction Networks

e.g.

circuit

terrorist organization



Interaction Networks

e.g.

circuit

terrorist organization

protein network

Interaction Networks

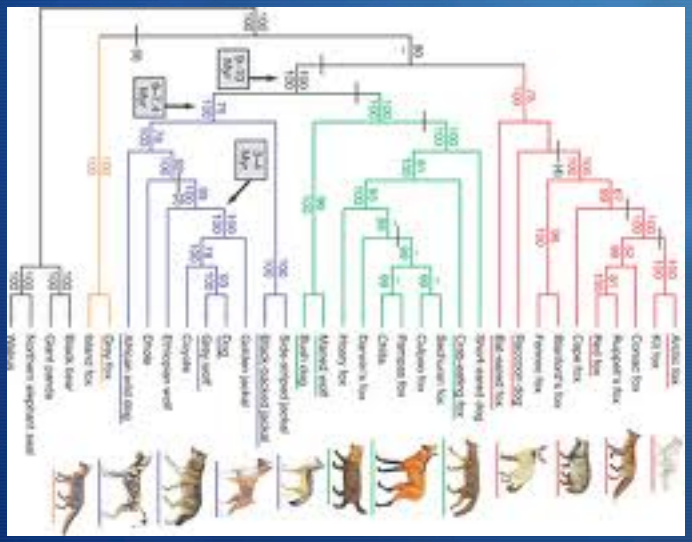
e.g.

circuit

terrorist organization

protein network

flock of geese



Phylogenetic Tree Reconstruction

[R-Srivastava '07a]

Problem: given different species, build their evolutionary tree.

d=0.01

Phylogenetic Tree Reconstruction

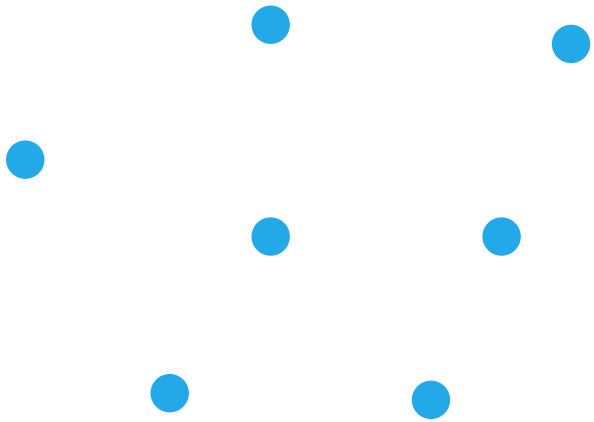
[R-Srivastava '07a]

Problem: given different species, build their evolutionary tree.

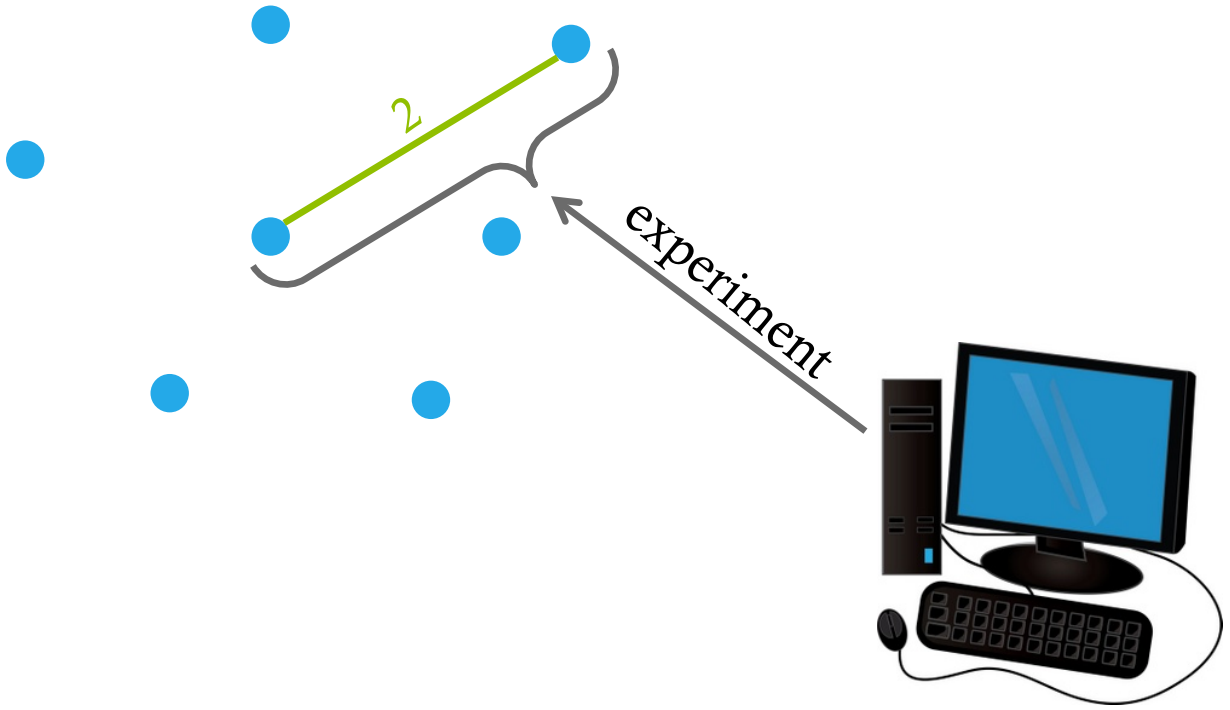
Query: can test genetic distance of a pair.

Property: distances are tree-realizable.

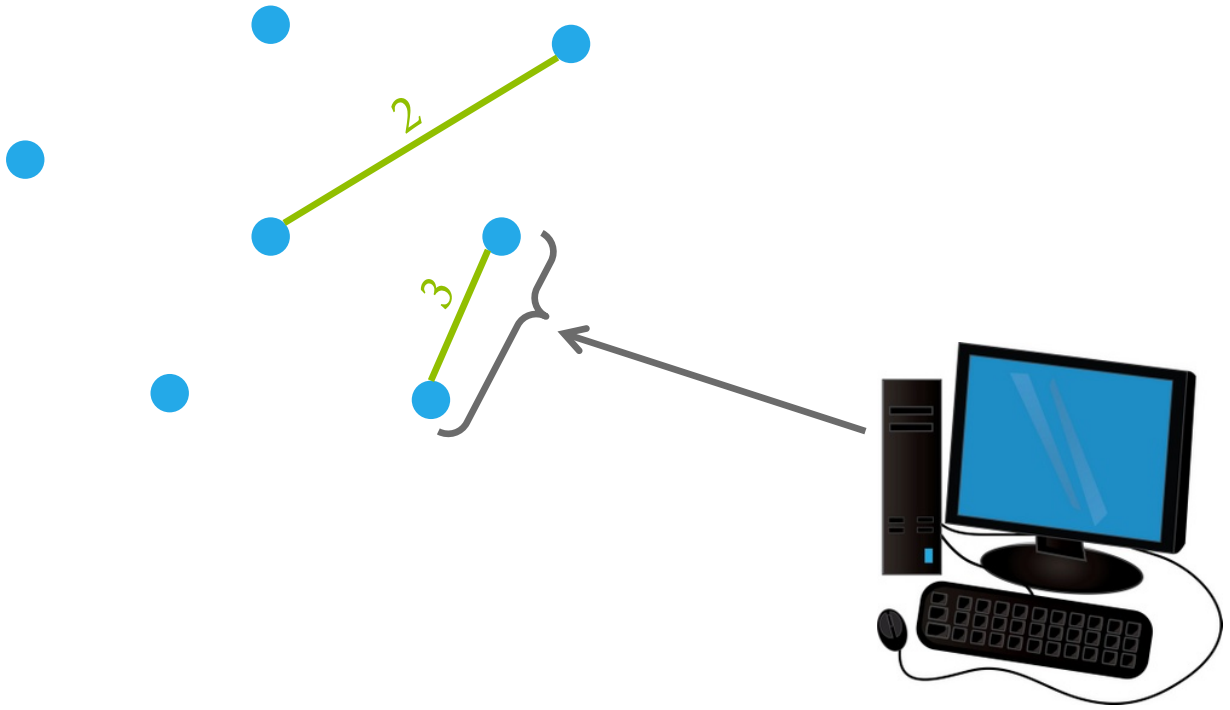
Graph Formulation



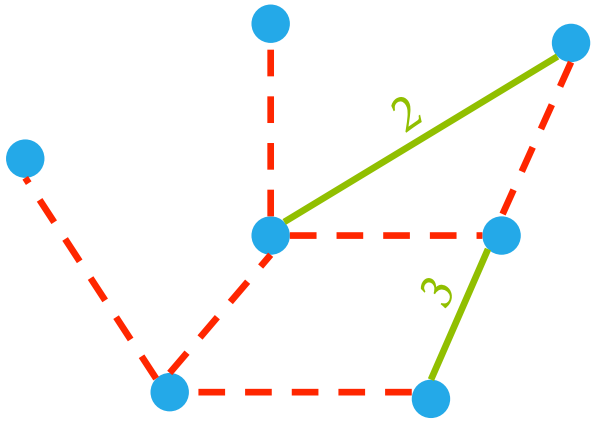
Graph Formulation



Graph Formulation

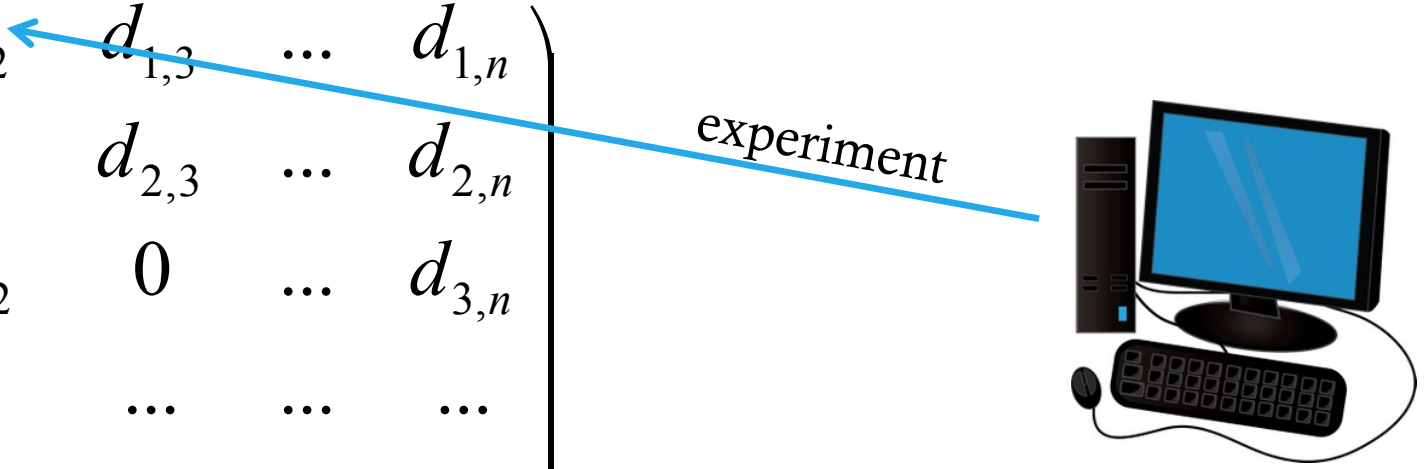


Graph Formulation



Matrix Formulation

$$\begin{pmatrix} 0 & d_{1,2} & d_{1,3} & \dots & d_{1,n} \\ d_{2,1} & 0 & d_{2,3} & \dots & d_{2,n} \\ d_{3,1} & d_{3,2} & 0 & \dots & d_{3,n} \\ \dots & \dots & \dots & \dots & \dots \\ d_{n,1} & d_{n,2} & d_{n,3} & \dots & 0 \end{pmatrix}$$



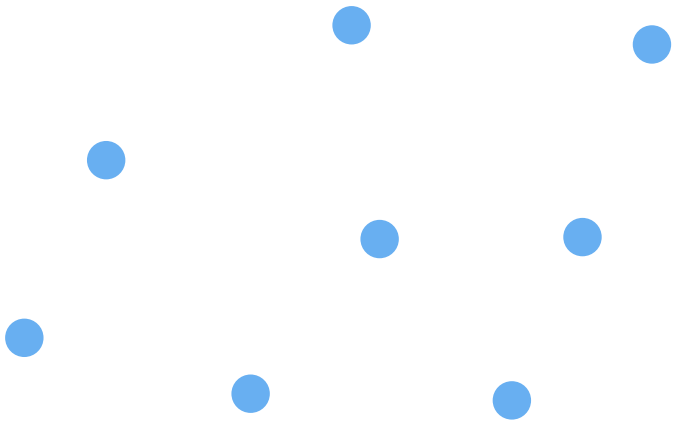
Problem also known as "distance matrix reconstruction."

Tree Reconstruction

- For n species, trivial number of queries is n^2.
- [Hein '89] gave algorithm achieving O(dnlogn).
  - Known to be optimal [King et al. '03].
- Yet a widely used algorithm was **`Longest-Path`** [Culberson-Rudnicki '89].
  - It was believed to also run in time O(dnlogn).
- We [**R**-Srivasatava '07] gave the first correct analysis of **`Longest-Path`**.

Longest-Path

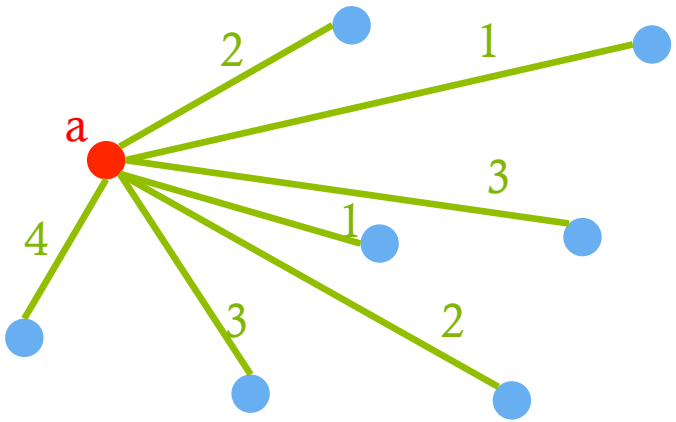
key insight: a longest path through a tree can be found using 3n queries.



Longest-Path

key insight: a longest path through a tree can be found using 3n queries.

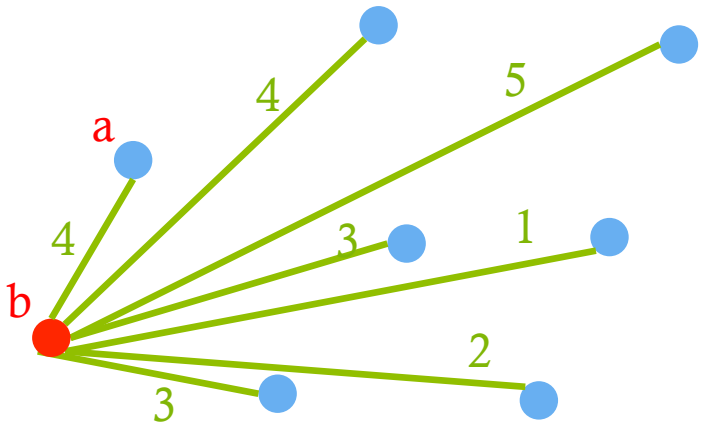
1. Pick any node a, query all distances from it.



Longest-Path

key insight: a longest path through a tree can be found using 3n queries.

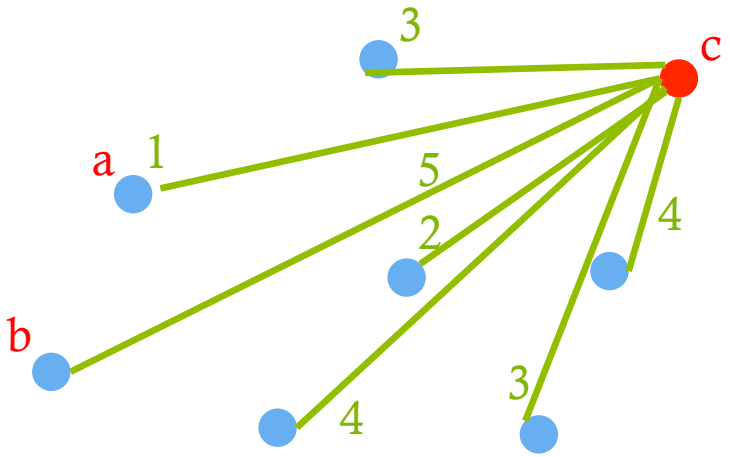
1. Pick any node a, query all distances from it.
2. Query all distances from b, the farthest node from a.



Longest-Path

key insight: a longest path through a tree can be found using 3n queries.

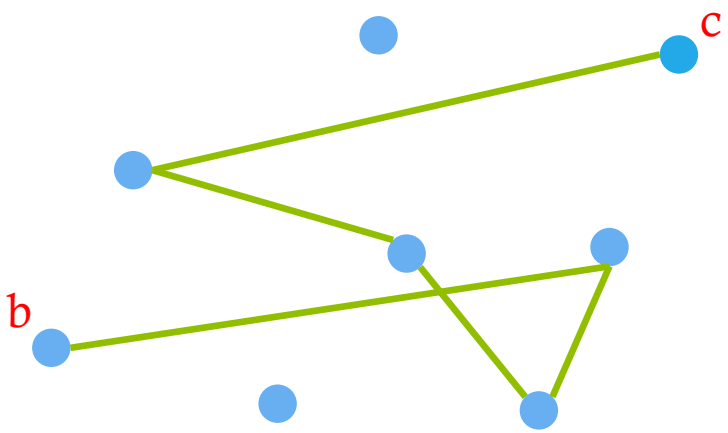
1. Pick any node a, query all distances from it.
2. Query all distances from b, the farthest node from a.
3. Query all distances from c, the farthest node from b.



Longest-Path

key insight: a longest path through a tree can be found using 3n queries.

1. Pick any node a, query all distances from it.
2. Query all distances from b, the farthest node from a.
3. Query all distances from c, the farthest node from b.

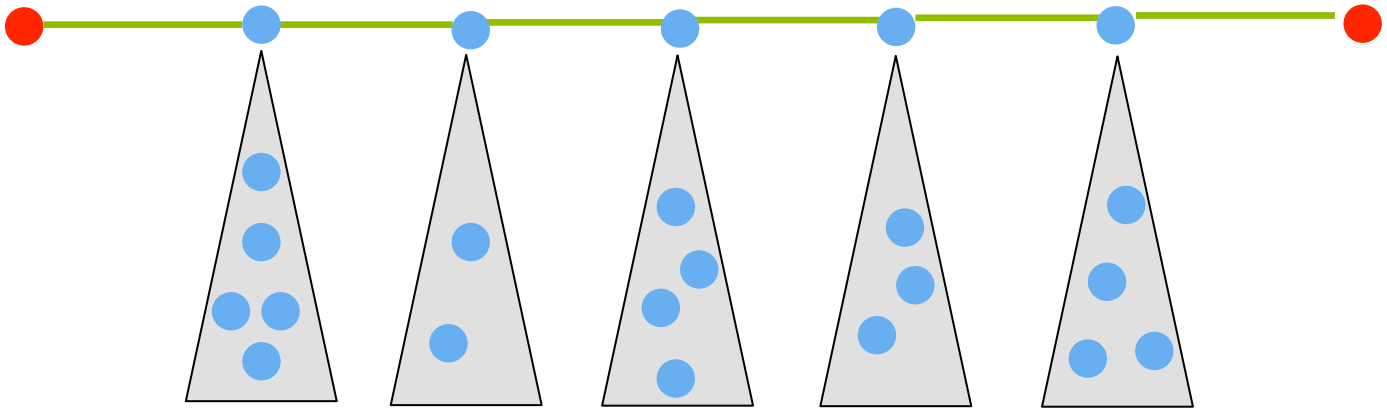


This is enough to reconstruct the longest path from b to c.

Longest-Path

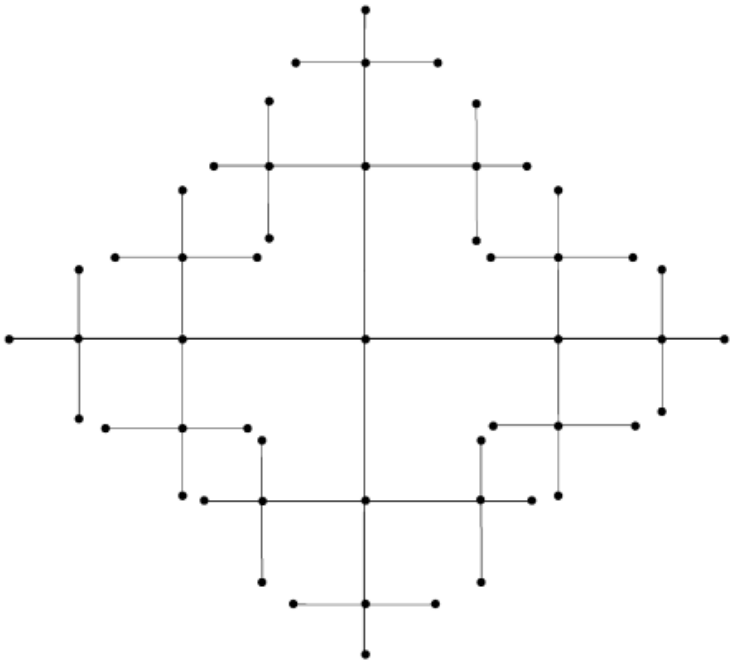
Algorithm

1. Find longest path.
2. Recurse on sub-trees.

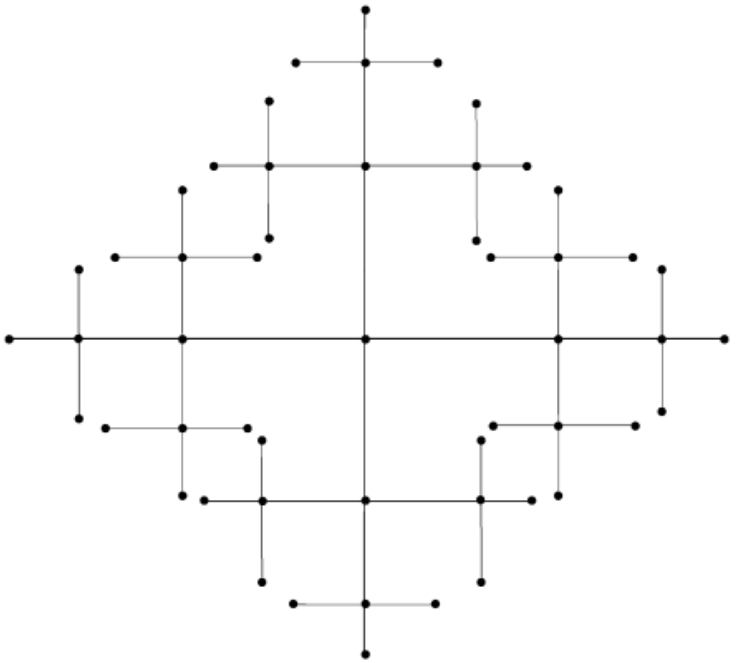


bad case [Culberson-Rudnicki '89]

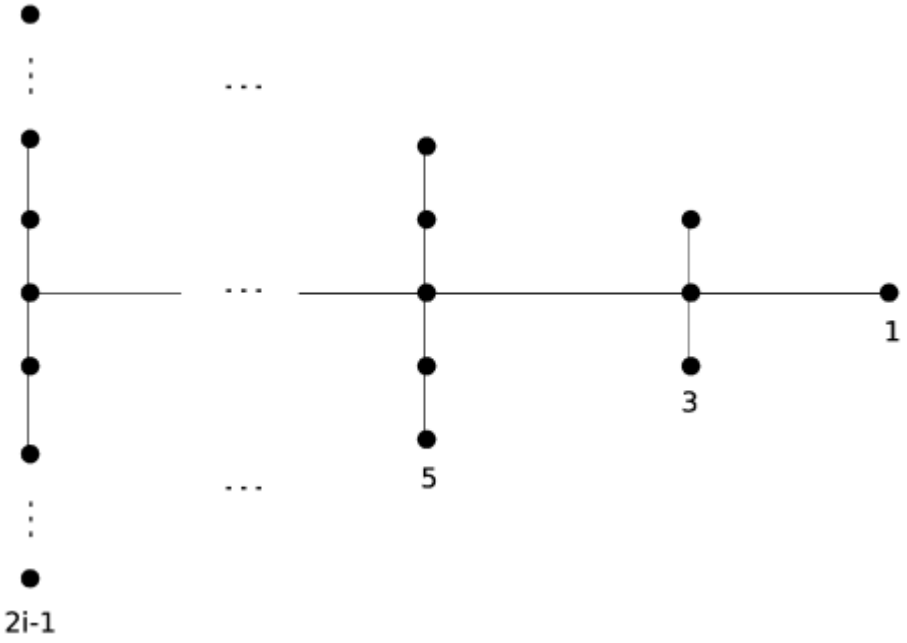
`Longest-Path` takes $\Omega(dn \log n)$



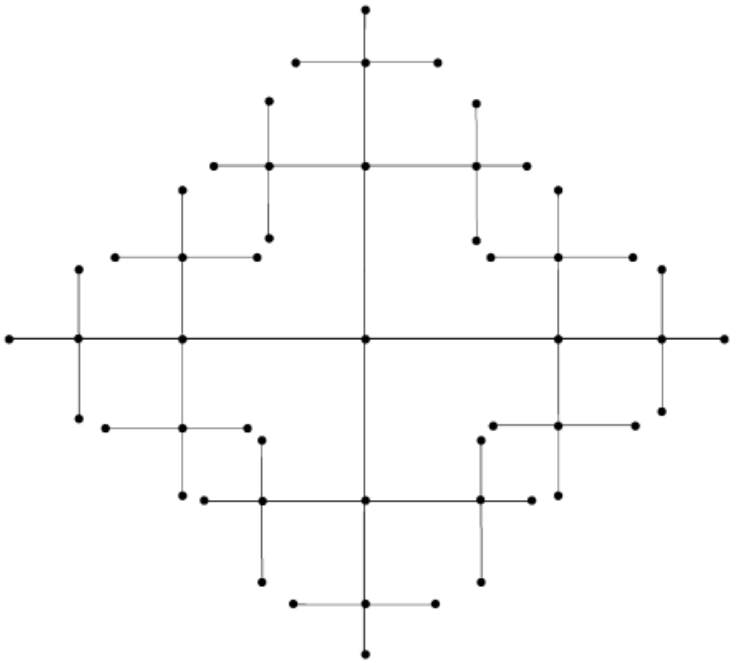
bad case [Culberson-Rudnicki '89]
`Longest-Path` takes $\Omega(dn\log n)$



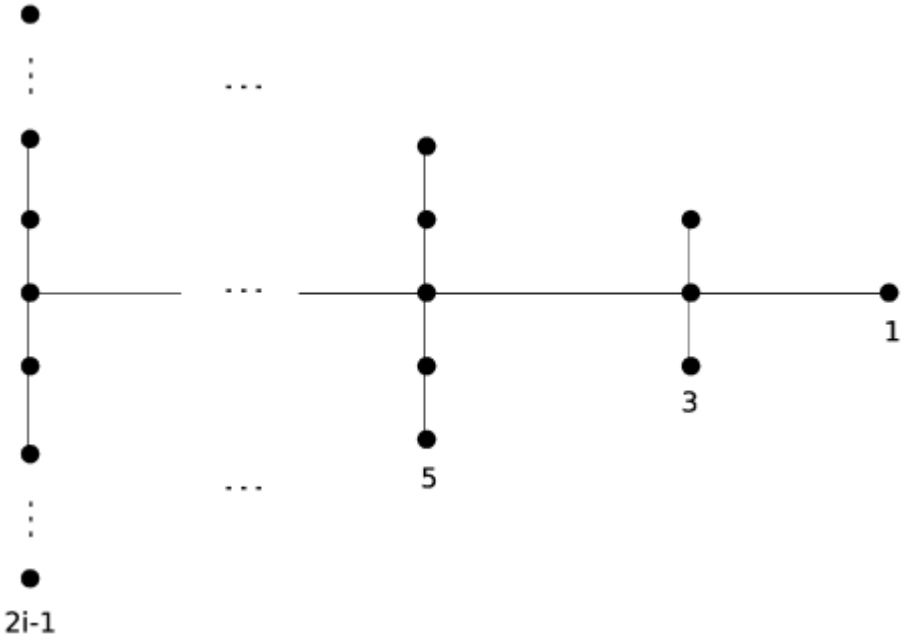
even worse case [R-Srivatava '07a]
`Longest-Path` takes $\Omega(n^{3/2}d^{1/2})$



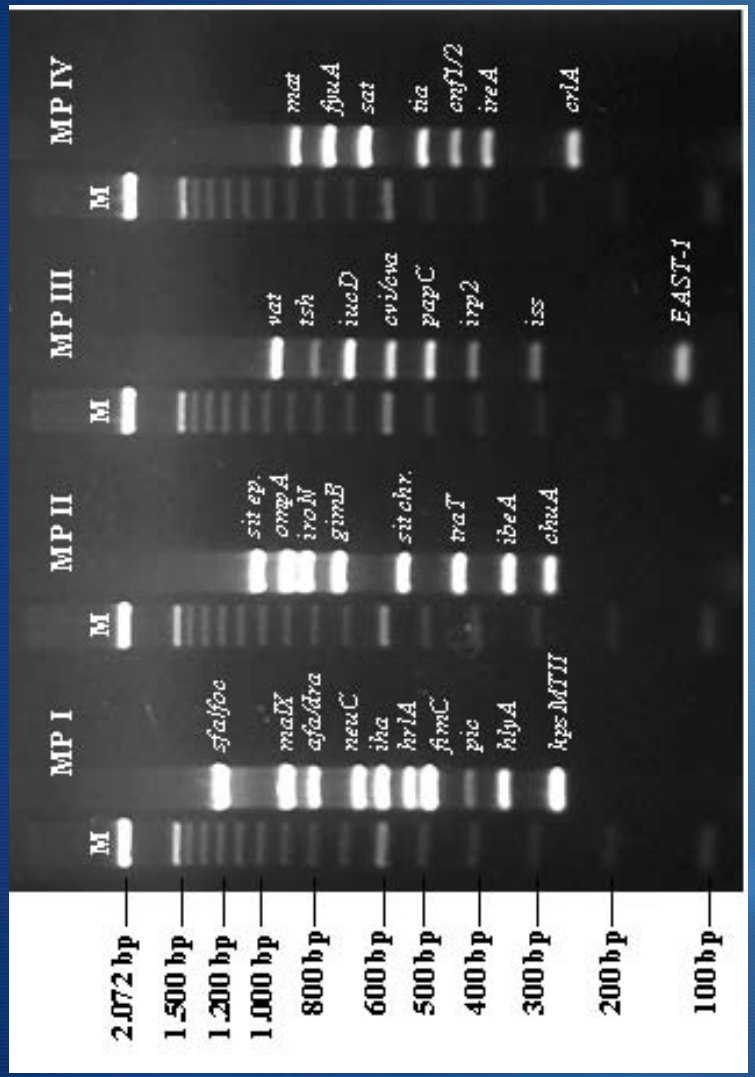
bad case [Culberson-Rudnicki '89]
`Longest-Path` takes $\Omega(dn\log n)$



even worse case [**R**-Srivatava '07a]
`Longest-Path` takes $\Omega(n^{3/2}d^{1/2})$



Main Theorem [**R**-Srivastava '07a]: **`Longest-Path`** runs in time $O(n^{3/2}d^{1/2})$ on topological trees.

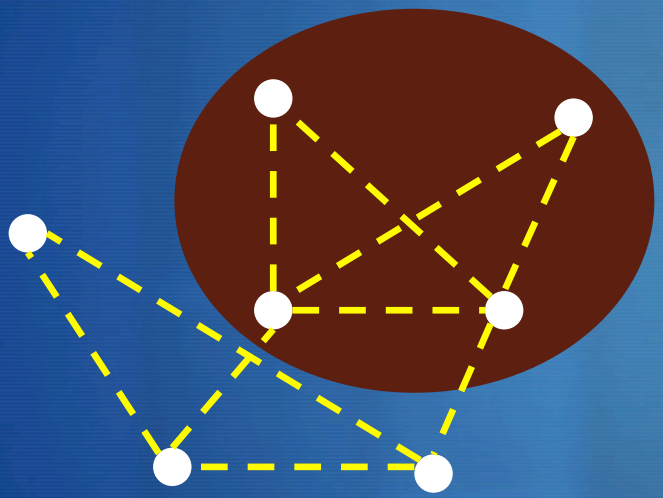


Genome Sequencing

[R-Srivastava '07b]

Problem: determine relative placement of "contigs" in a genome.

Query: **Multiplex PCR** counts the number of contiguous pairs of primers.



Genome Sequencing

[R-Srivastava '07b]

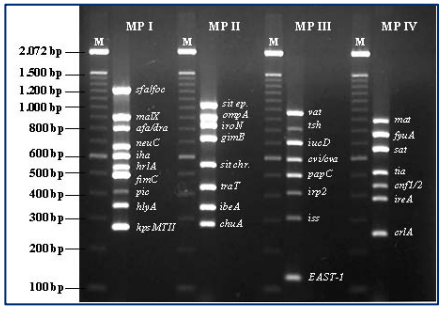
Problem: determine relative placement of "contigs" in a genome.

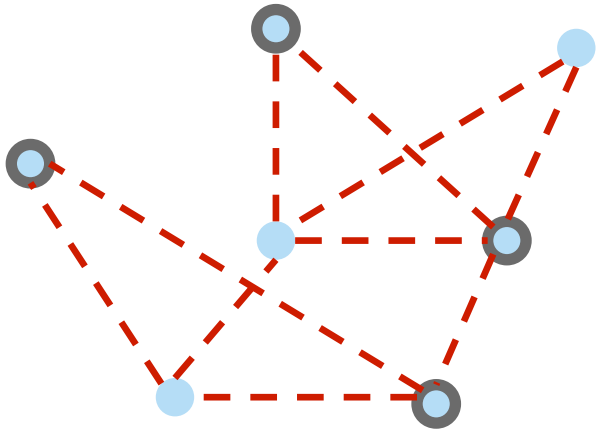
Query: **Multiplex PCR** counts the number of contiguous pairs of primers.

Reformulation: Learn an un-weighted graph by "edge counting." (Additive model in bioinformatics.)

EC model: un-weighted graph hidden from learner
learner can ask "edge counting queries"

goal: reconstruct graph

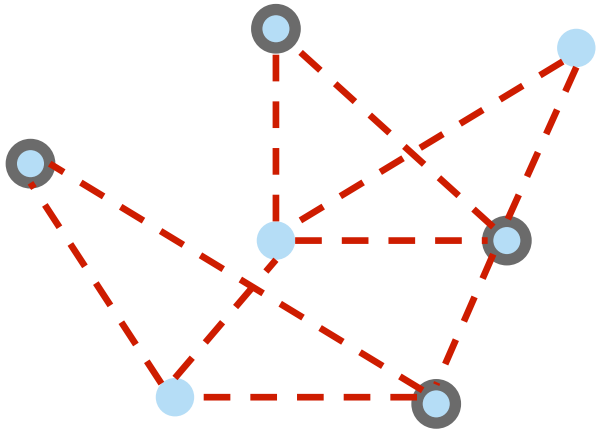




Edge counting (EC) query: learner picks subset of vertices and is told the number of edges on their induced subgraph.

ED model: un-weighted graph hidden from learner
learner can ask "edge detecting queries"

goal: reconstruct graph



Edge counting (ED) query: learner picks subset of vertices and is told whether their induced subgraph is an independent set.

Previous Work

Edge Detecting (ED)

- Arbitrary Graphs
  [Angluin-Chen '04]
- Hidden Matchings
  [Alon et al. '04]
- Hamiltonian Cycles
  [Grebinski-Kucherov '98]

Edge Counting (EC)

- trees, degree bounded graphs
  [Grebinski-Kucherov '00]
- optimal algorithm (exp time)
  [Choi-Kim '08]
- k-degenerate graphs
  [Bouvel et al. '05]

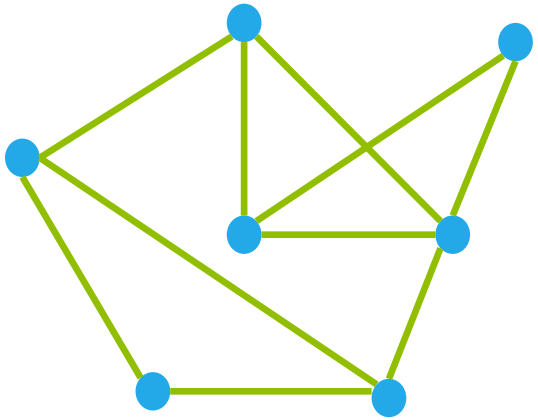
Results for Poly-Time Learners

d = degree, k = number of components

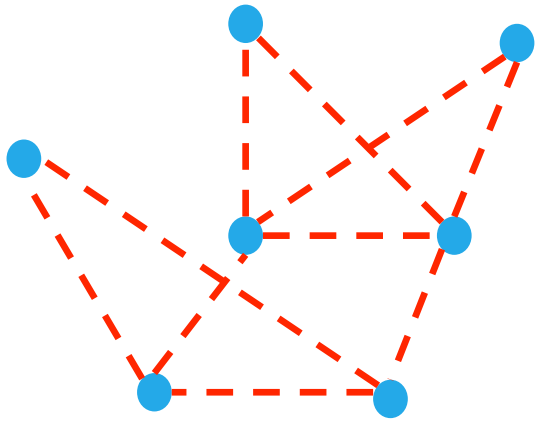
	Graph	tree	partition
ED Query	$\Theta(\|E\| \lg n)$, $\Theta(n^2)$ [Angluin-Chen '04]	$\Theta(n \lg n)$ [from ←]	$\Theta(n^2)$ [**R**-Srivastava '07b]
EC Query	$O(\|E\| \lg n)$ [from ↑] [**R**-Srivastava '07b] $\Theta(n^2/\lg n)$ $\Theta(dn)$ [Grebinski-Kucherov '00]	$\Theta(n)$ [Grebinski-Kucherov '00]	$O(n \lg n)$ $\Omega(n)$ [**R**-Srivastava '07b]
distance Query	$\Theta(n^2)$ [from →]	$\Theta(n^2)$ [**R**-Srivastava '07b] $O(dn \lg n)$ [Hein '89]	$\Theta(nk)$ [**R**-Srivastava '07b]

Verification
a task easier than learning

Verifier presented with graph H and given EC query access to G.

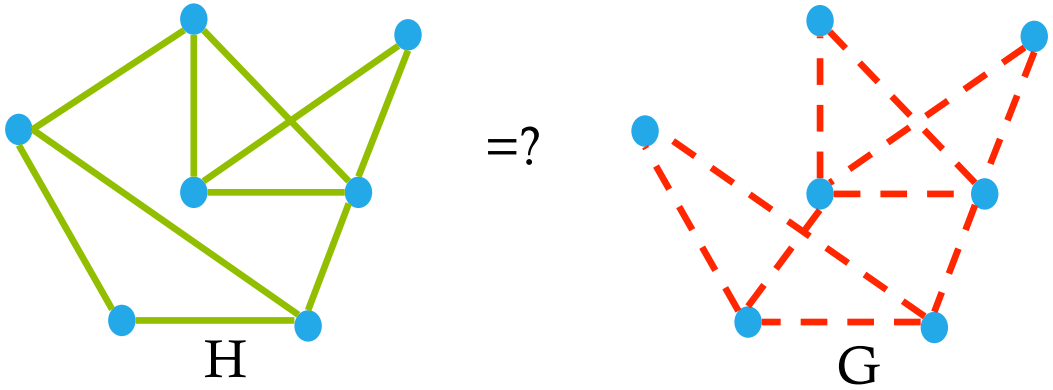


H

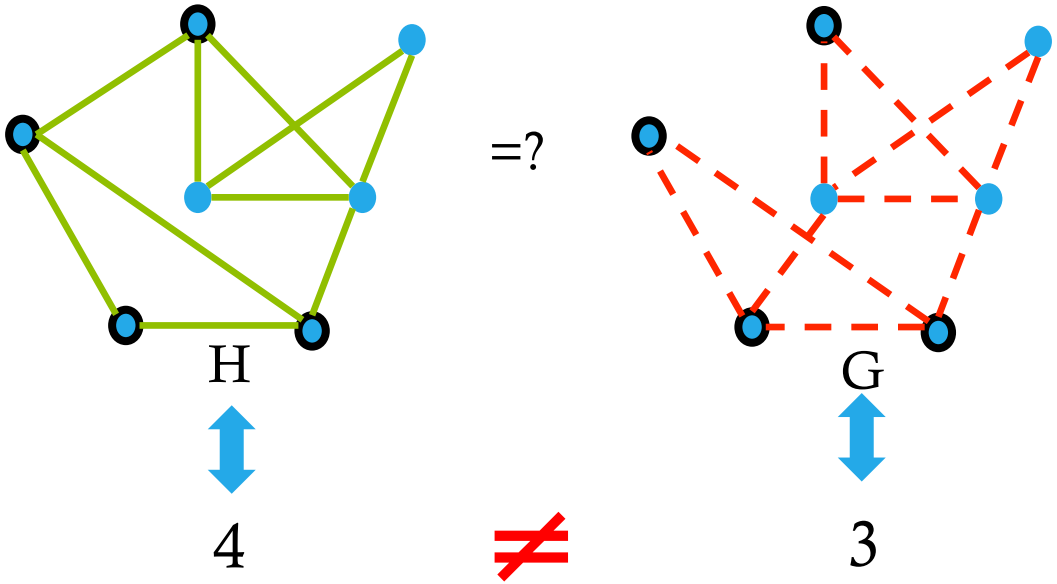


G

Verifier needs to answer: does H = G?



Theorem [**R**-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.



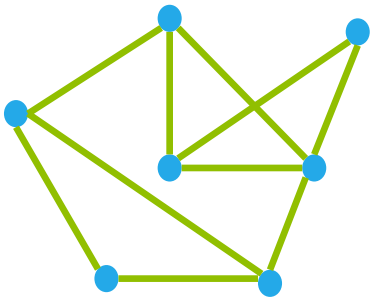
Theorem [R-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

Theorem [**R**-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

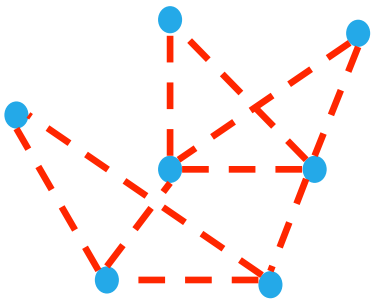
Lemma: A random subset of vertices of a non-empty graph induces an **odd** number of edges with probability ≥ ¼.

Theorem [**R**-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

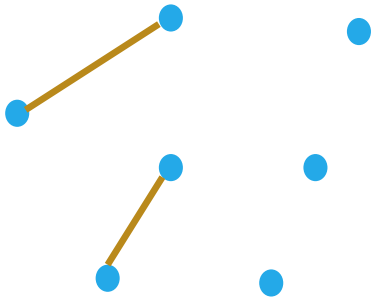
Lemma: A random subset of vertices of a non-empty graph induces an **odd** number of edges with probability ≥ ¼.



H



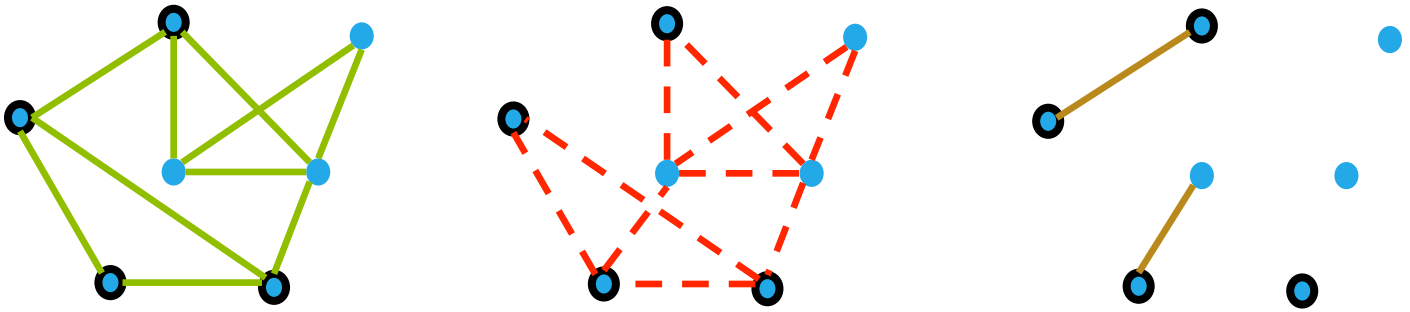
G



H△G

Theorem [**R**-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

Lemma: A random subset of vertices of a non-empty graph induces an **odd** number of edges with probability ≥ ¼.



Theorem [**R**-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

Lemma: A random subset of vertices of a non-empty graph induces an **odd** number of edges with probability ≥ ¼.

Theorem [**R**-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

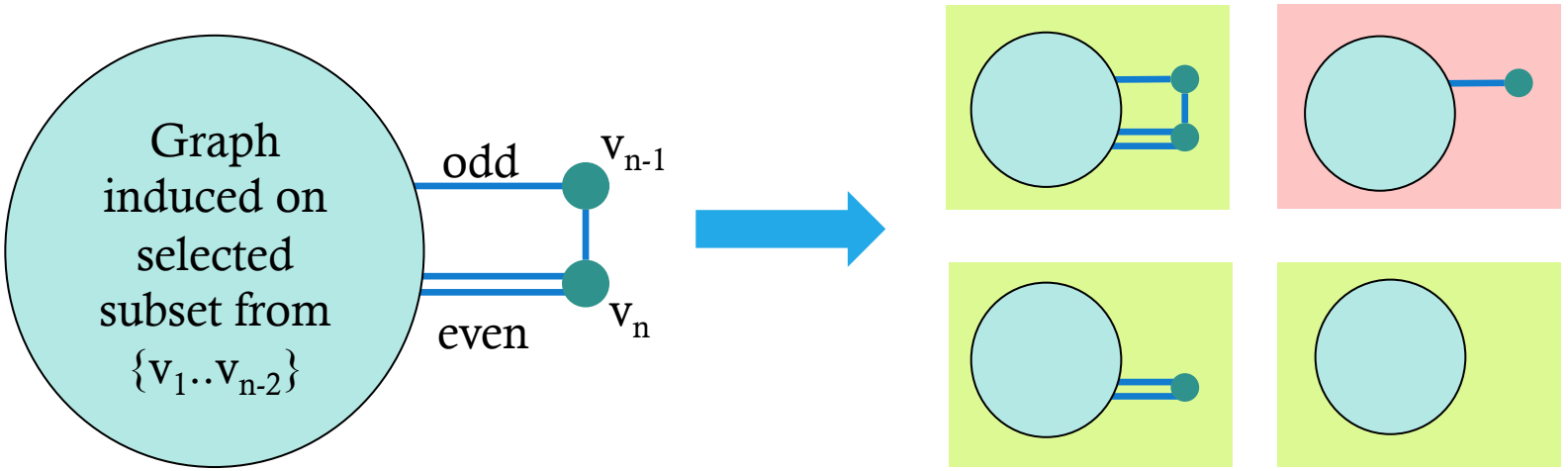
Lemma: A random subset of vertices of a non-empty graph induces an **odd** number of edges with probability ≥ ¼.

we choose the random subset as follows:

1. order the vertices $v_1 \ldots v_n$ such that (v_{n-1}, v_n) is an edge
2. in order, flip a fair coin for each vertex to decide if it's in the subset

Theorem [**R**-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

Lemma: A random subset of vertices of a non-empty graph induces an **odd** number of edges with probability ≥ ¼.



Theorem [R-Srivastava '07b]: If H ≠ G, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability ≥ ¼.

Any graph can be verified by a randomized algorithm with error ε using O(log(1/ε)) EC queries!

Theorem [R-Srivastava '07b]: If $H \neq G$, querying a random subset of vertices on G and simulating the same query on H will produce different answers with probability $\geq \frac{1}{4}$.

Any graph can be verified by a randomized algorithm with error ε using $O(\log(1/\varepsilon))$ EC queries!

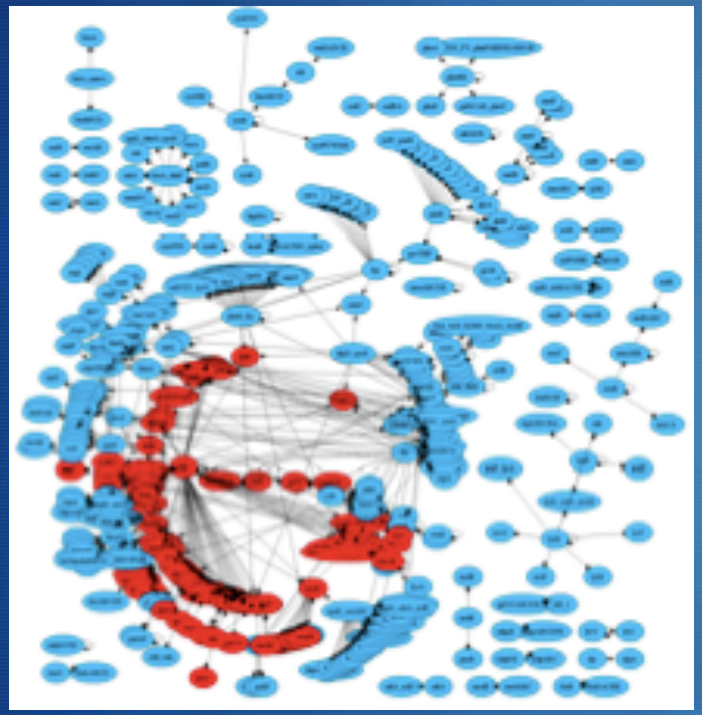
Matrix formulation

A: n×n adjacency matrix

x: $\{0,1\}^n$ is query vector

EC query = $x^T Ax$

Our result gives an improved technique for matrix fingerprinting [Freivalds '77] using less randomness.



Gene Regulatory Networks

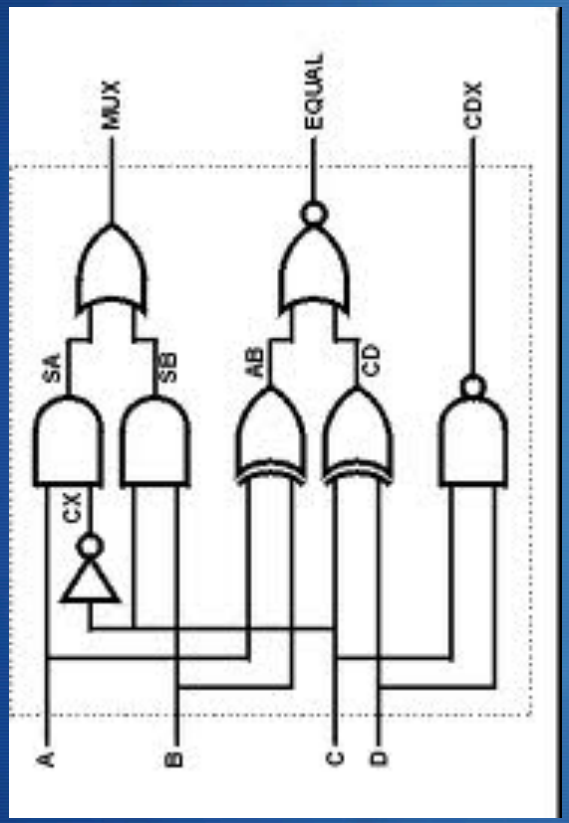
[Angluin-Aspnes-Chen-**R** '07]
[Angluin-Aspnes-Chen-Eisenstat-**R** '08]

Problem: A GRN is a collection of genes interacting according to unknown rules. The goal is to learn their topology and function.

Query: The genes can be disrupted or expressed by the learner. Only the final phenotype can be observed.

Gene Regulatory Networks

[Angluin-Aspnes-Chen-**R** '07]
[Angluin-Aspnes-Chen-Eisenstat-**R** '08]



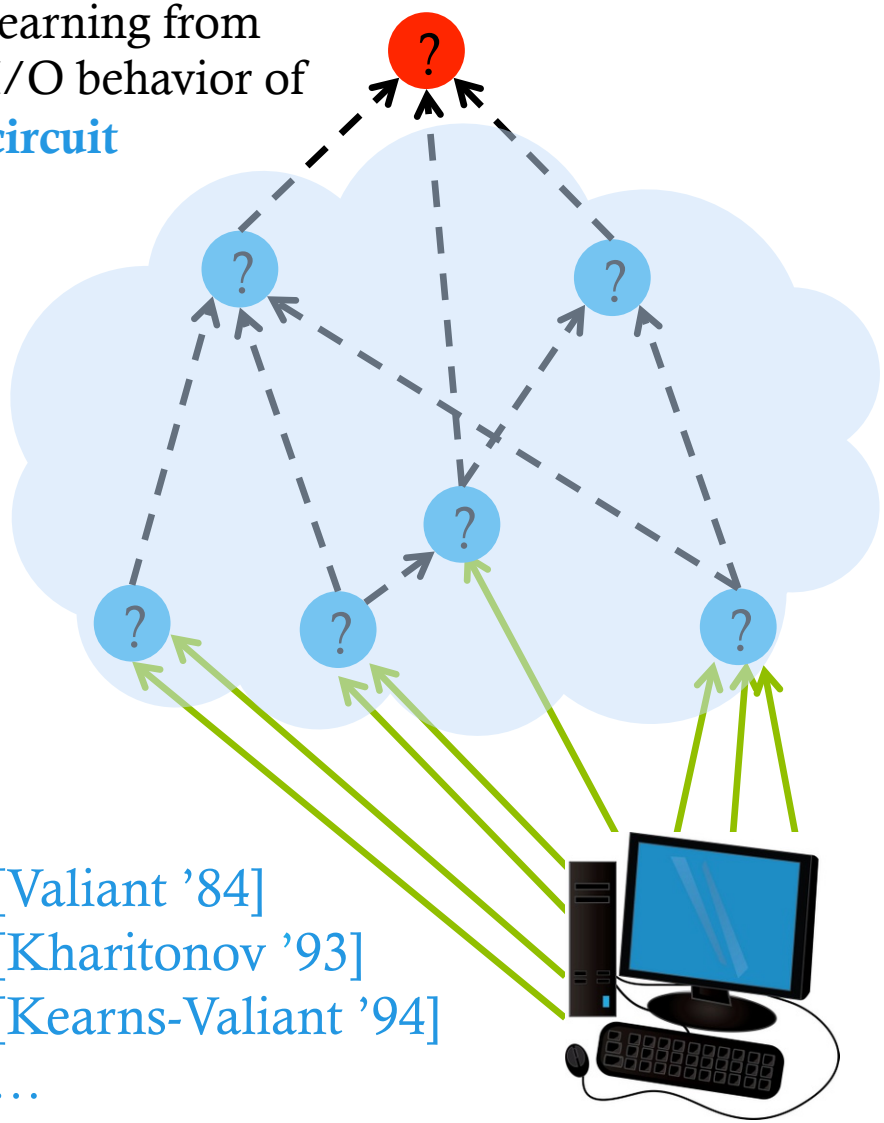
Problem: A GRN is a collection of genes interacting according to unknown rules. The goal is to learn their topology and function.

Query: The genes can be disrupted or expressed by the learner. Only the final phenotype can be observed.

Reformulation: Learning analog/ probabilistic circuits by injecting values.

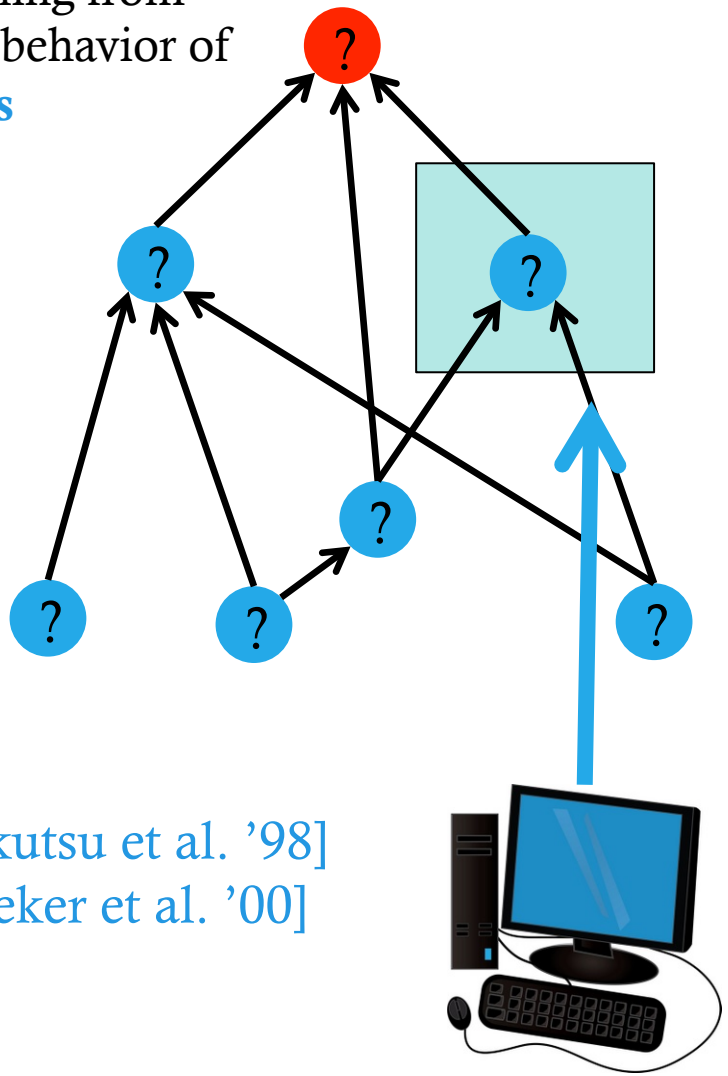
Two Models of Learning Circuits

learning from
I/O behavior of
circuit



[Valiant '84]
[Kharitonov '93]
[Kearns-Valiant '94]
…

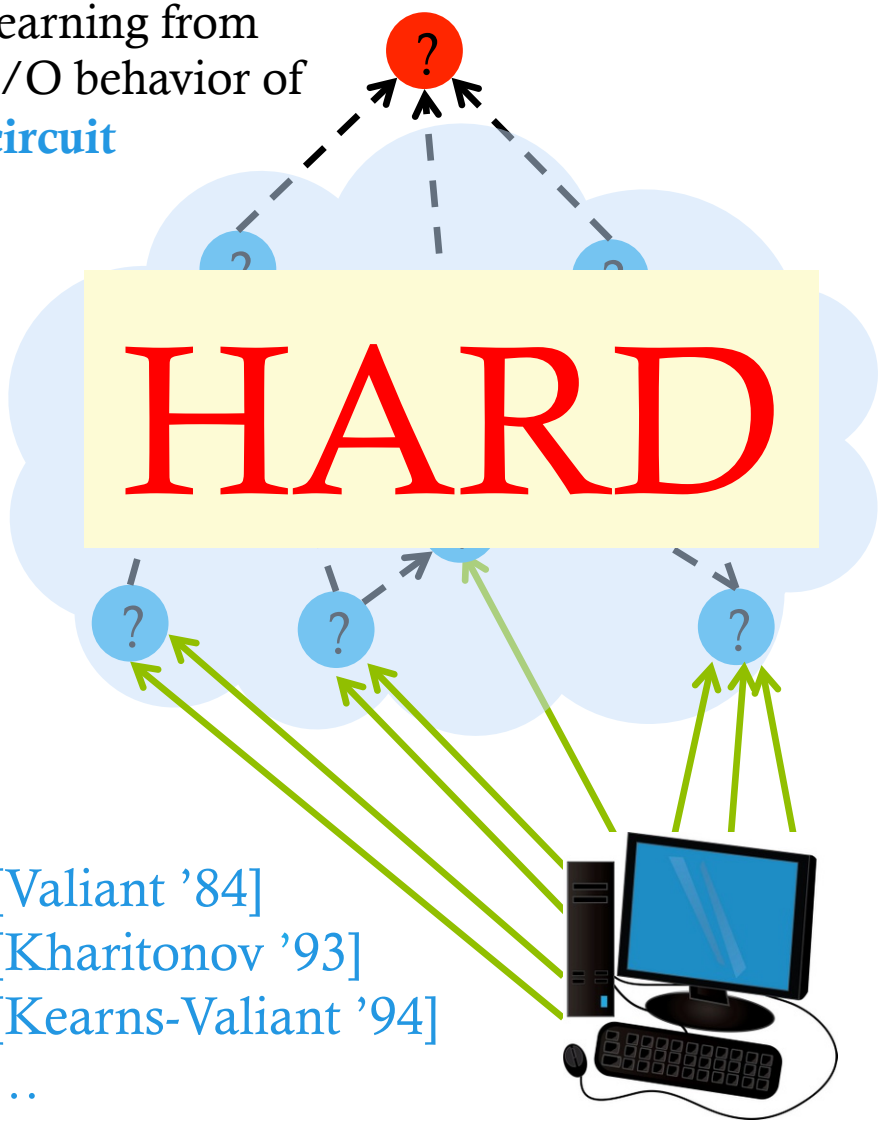
learning from
I/O behavior of
gates



[Akutsu et al. '98]
[Ideker et al. '00]

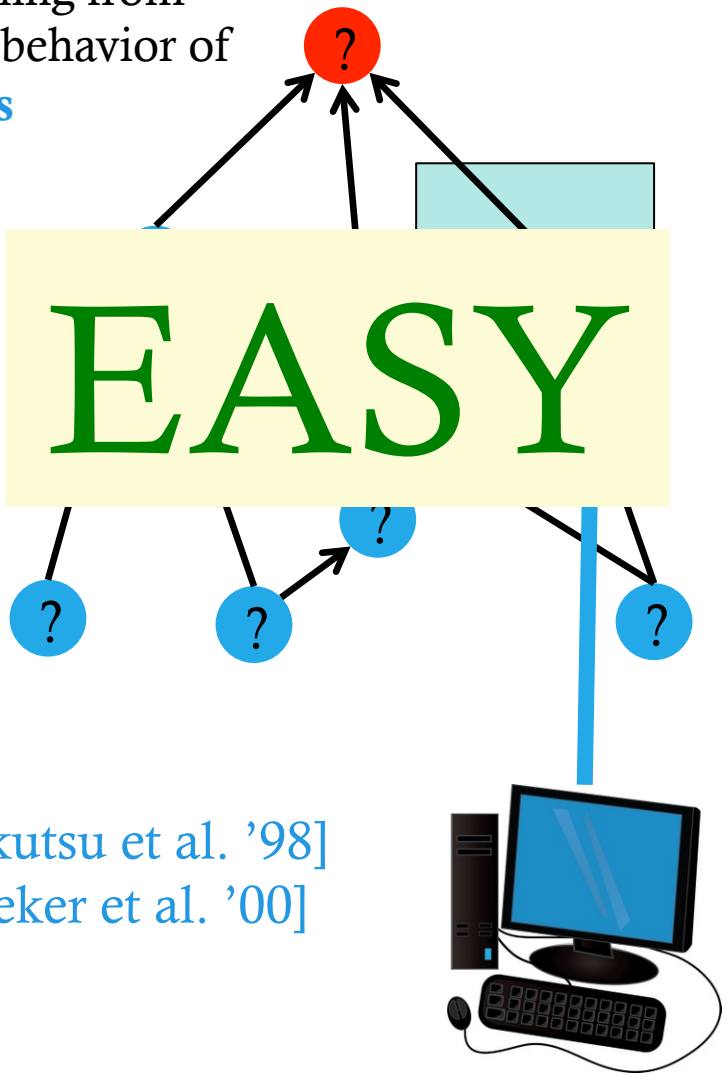
Two Models of Learning Circuits

learning from
I/O behavior of
circuit



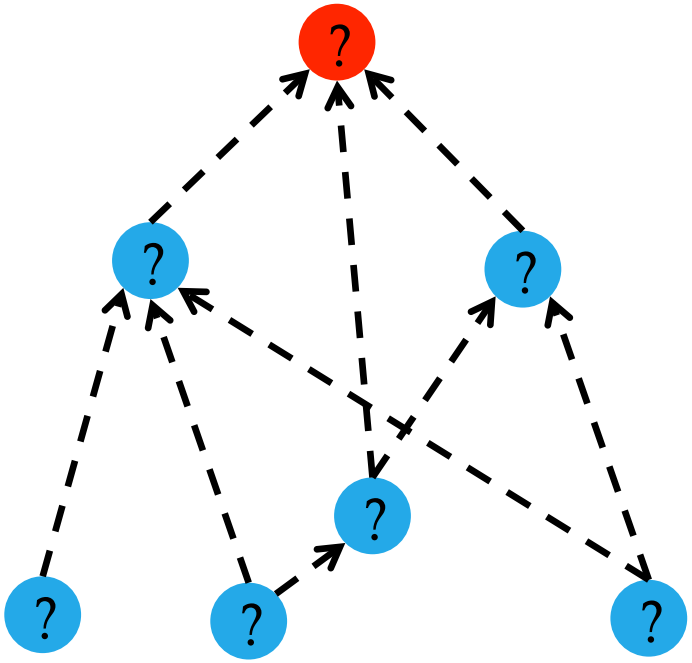
[Valiant '84]
[Kharitonov '93]
[Kearns-Valiant '94]
…

learning from
I/O behavior of
gates



[Akutsu et al. '98]
[Ideker et al. '00]

Learning by Injecting Values

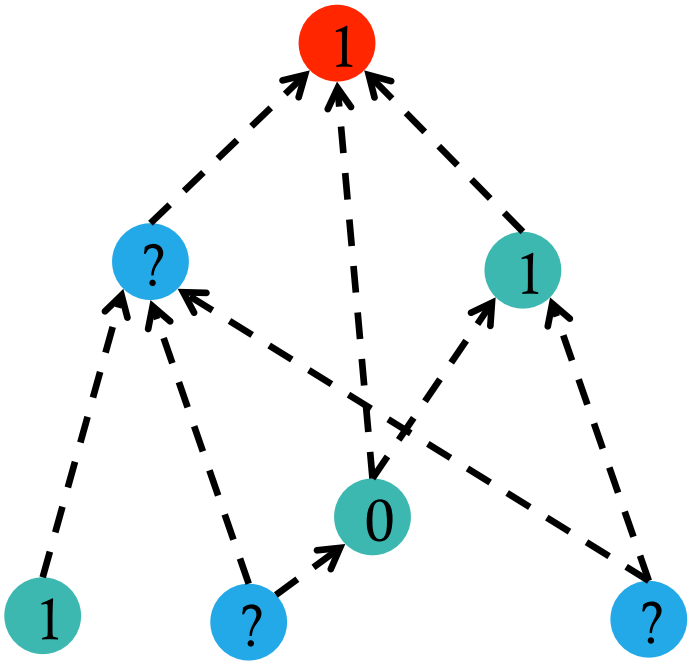


Value Injection Query Model
[Angluin et al. '06]

Learner does not know how the gates are connected or what the gates compute.

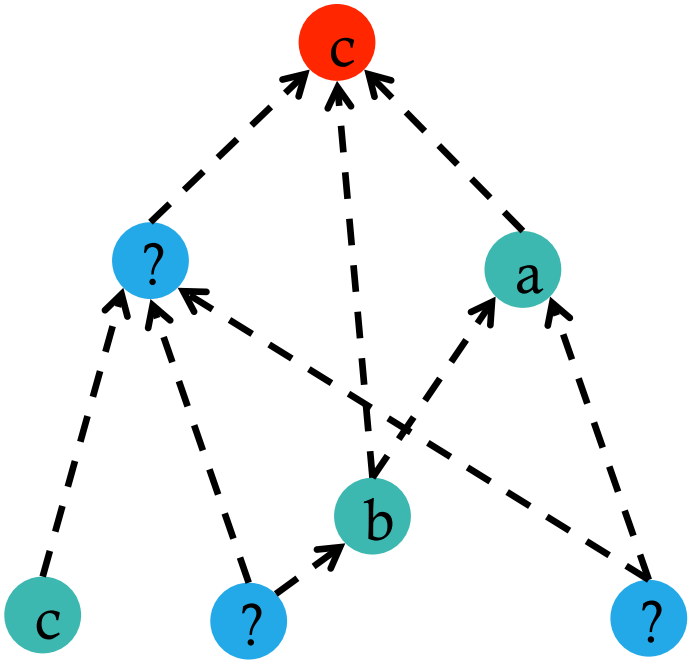
Learner can override the values on gates and observe what happens at the output.

Learning by Injecting Values



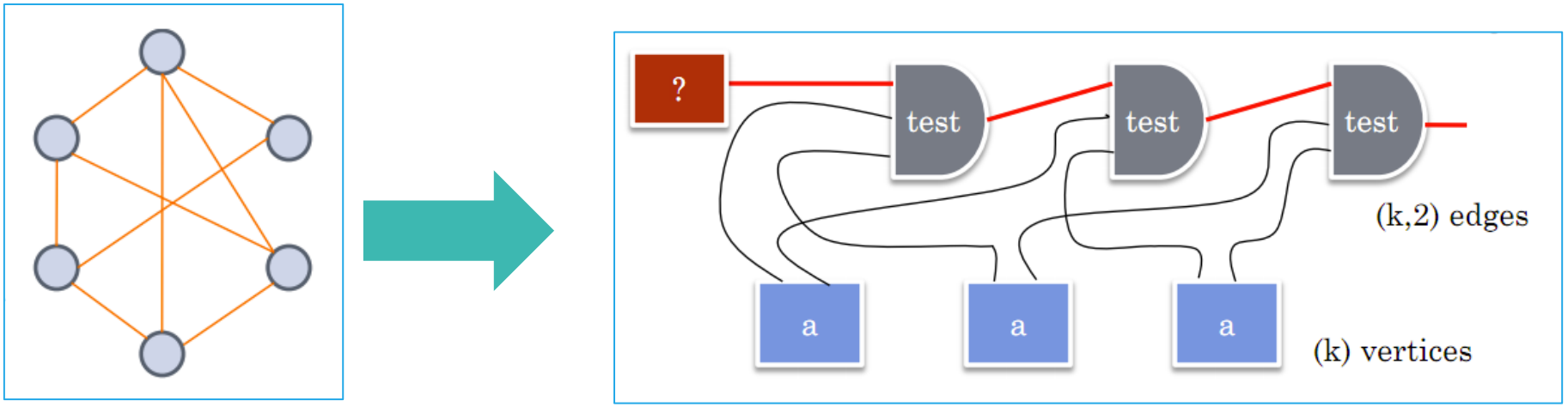
Value Injection Query Model
[Angluin et al. '06]

Learning by Injecting Values



We considered large-alphabet and analog circuits.

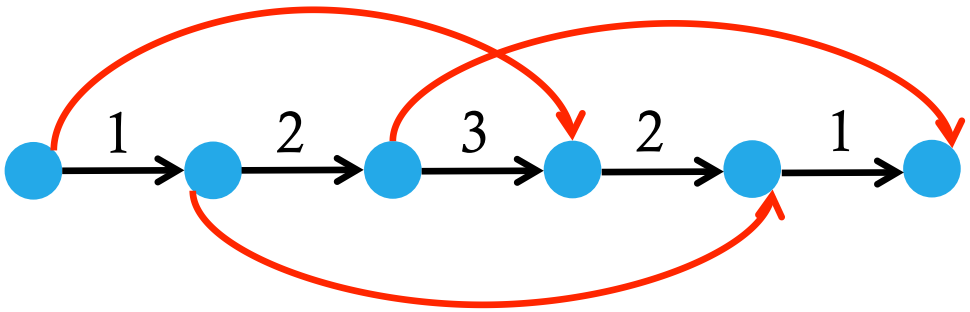
Theorem [Angluin-Aspnes-Chen-**R** '07]: An algorithm for learning bounded fan-in circuits, polynomial in the number of wires and alphabet size would imply fixed parameter tractability of all problems in **W[1]**.



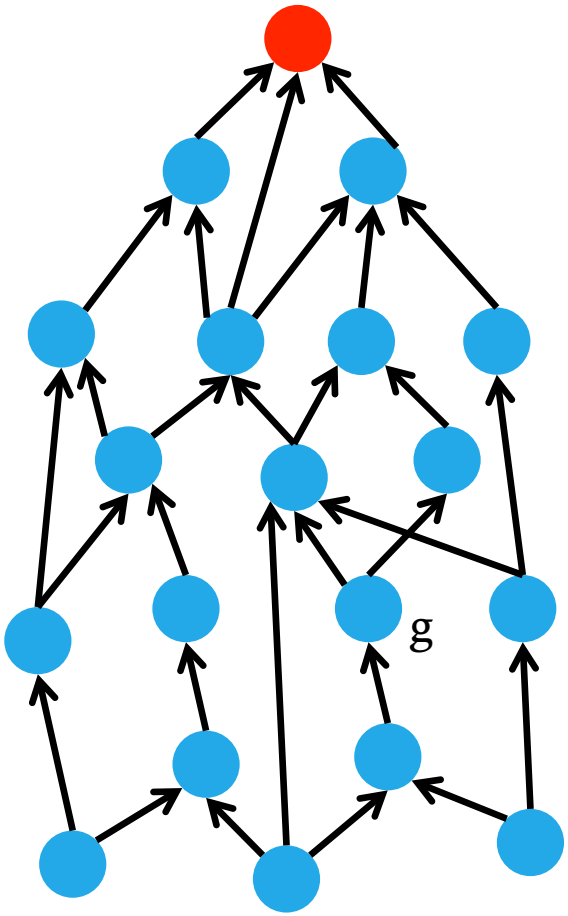
Main Theorem [Angluin-Aspnes-Chen-**R** '07]: The class of circuits having n wires, alphabet size s, fan-in bound k, and shortcut width bounded by b, is learnable using $ns^{O(k+b)}$ value injection queries and time polynomial in the number of queries.

Main Theorem [Angluin-Aspnes-Chen-**R** '07]: The class of circuits having n wires, alphabet size s, fan-in bound k, and shortcut width bounded by b, is learnable using $ns^{O(k+b)}$ value injection queries and time polynomial in the number of queries.

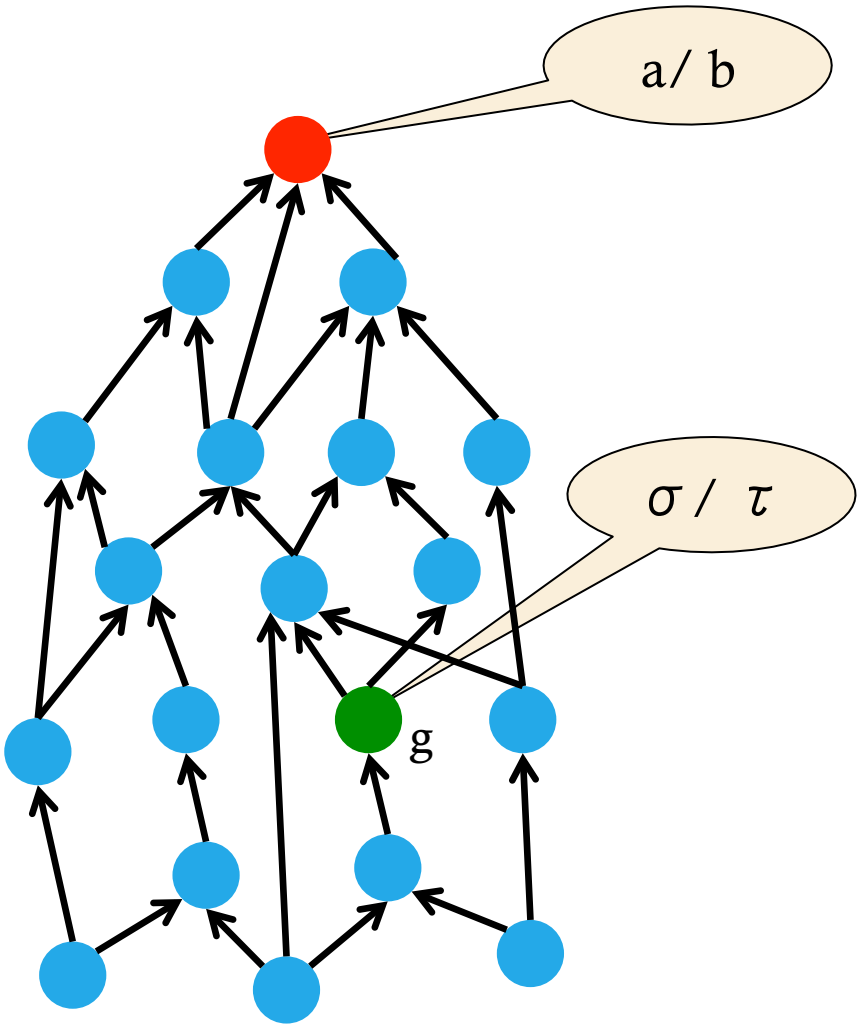
In some sense, this meets the W[1] hardness lower bound.



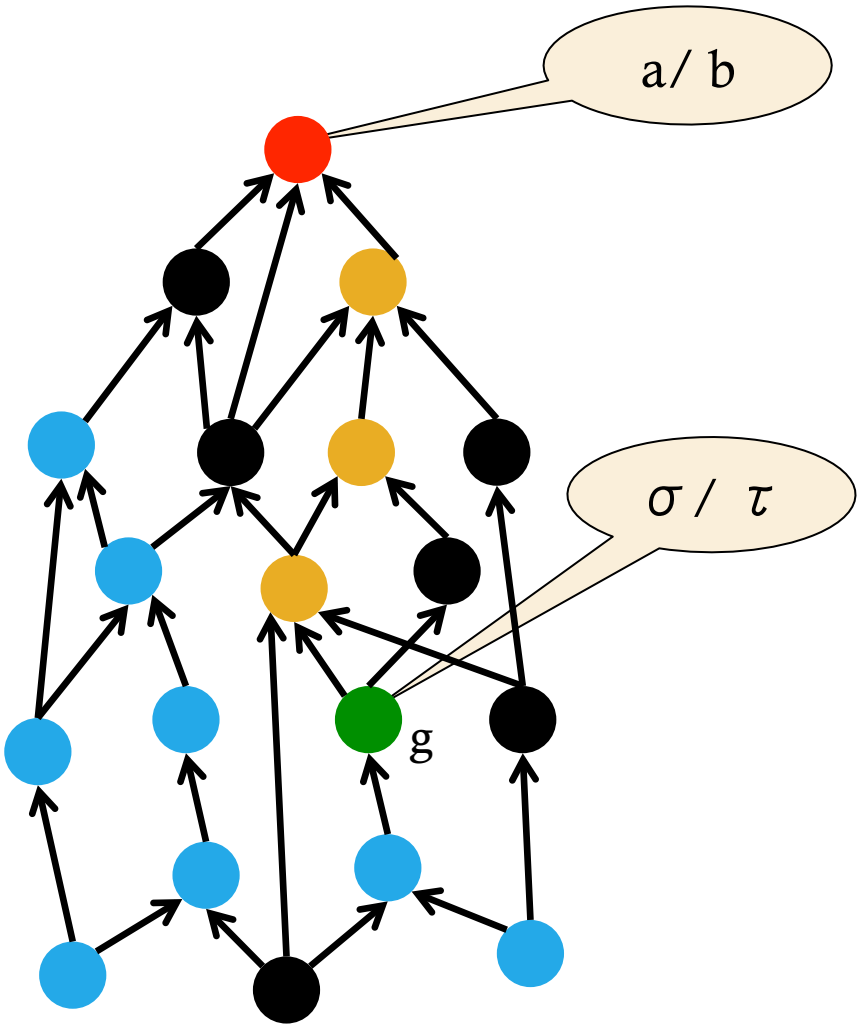
shortcut width = 3



Distinguishing Paths Lemma [Angluin-Aspnes-Chen-**R** '07]: Suppose σ and τ are distinguishable for gate g. Then … there is a distinguishing path π for gate g and values σ and τ.



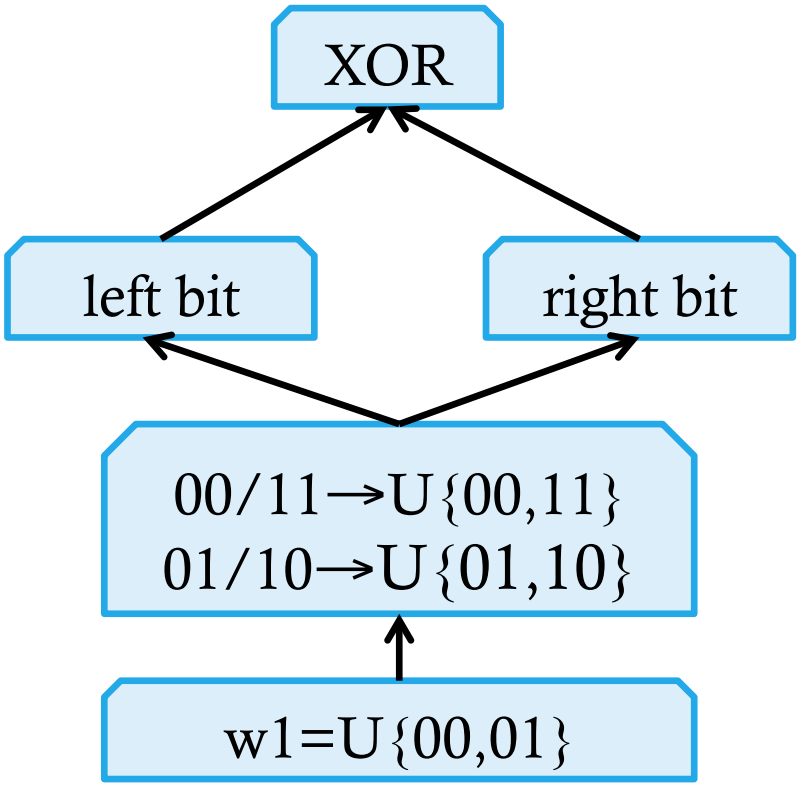
Distinguishing Paths Lemma [Angluin-Aspnes-Chen-**R** '07]: Suppose σ and τ are distinguishable for gate g. Then … there is a distinguishing path π for gate g and values σ and τ.

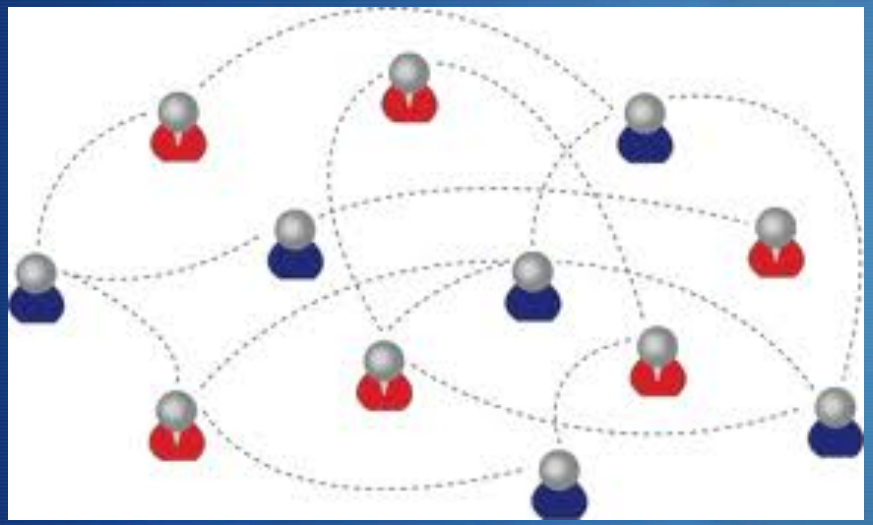


Distinguishing Paths Lemma [Angluin-Aspnes-Chen-**R** '07]: Suppose σ and τ are distinguishable for gate g. Then … there is a distinguishing path π for gate g and values σ and τ.

The Distinguishing Paths Lemma is not true for probabilistic circuits with alphabet size > 2!

[Angluin-Aspnes-Chen-Eisenstat-**R** '09]





Social Networks

[Angluin-Aspnes-**R** '08]

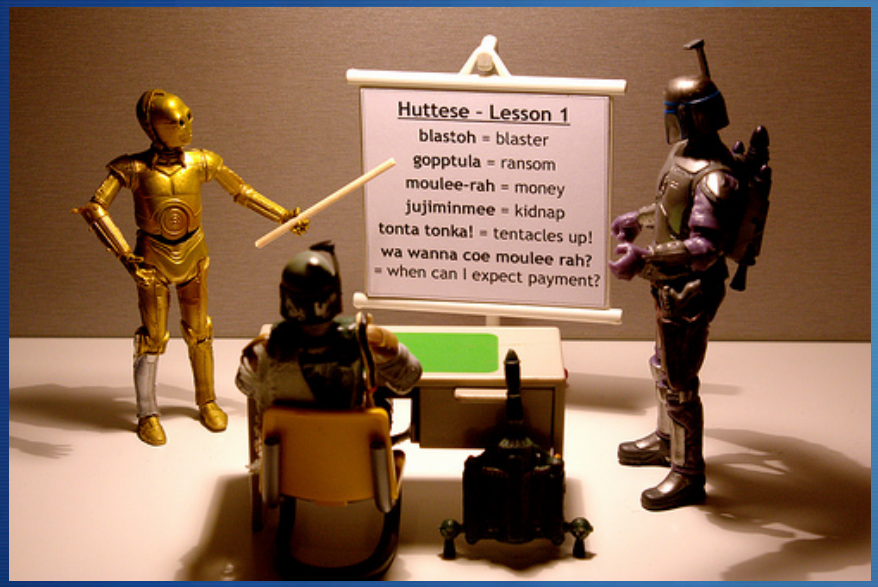
Problem: To learn the pairwise connections in an independent cascade social network.

Query: Nodes can be activated and suppressed. This corresponds to marketing, immunizing, etc.

Results: $\Theta(n^2)$ queries are needed.

bad news: large for large networks

good news: learn an edge per query.



Language Learning
[Angluin-Bonache-Dediu-**R** '08]

Problem: Learning a new language with the help of a teacher.

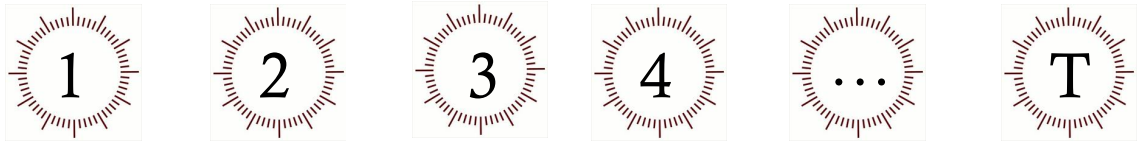
Query: The learner makes an utterance in a language and is then given information from a teacher.

Model: Learning labeled automata from membership queries.

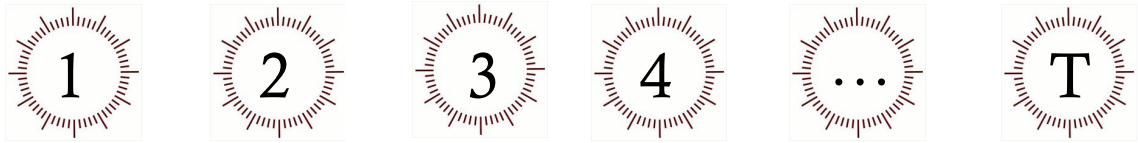
Multiarmed Bandits

My work at Yahoo! Research

Multiarmed Bandits
[Robbins '52]



Multiarmed Bandits
[Robbins '52]



1

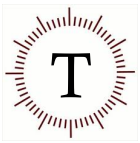
2 $0.50

3

⋮

k

Multiarmed Bandits
[Robbins '52]



1

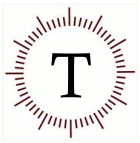
2 $0.50

3 $0

⋮

k

Multiarmed Bandits
[Robbins '52]



1 

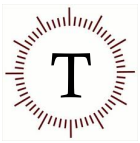
2  $0.50 $0.33

3  $0

⋮

k 

Multiarmed Bandits
[Robbins '52]



1 

$0.20

2 

$0.50 $0.33

3 

$0

⋮

k 

$0.90

Multiarmed Bandits
[Robbins '52]

1  → $0.5T

2  → $0.2T

3  → $0.33T

⋮

k  → $0.1T

Multiarmed Bandits
[Robbins '52]

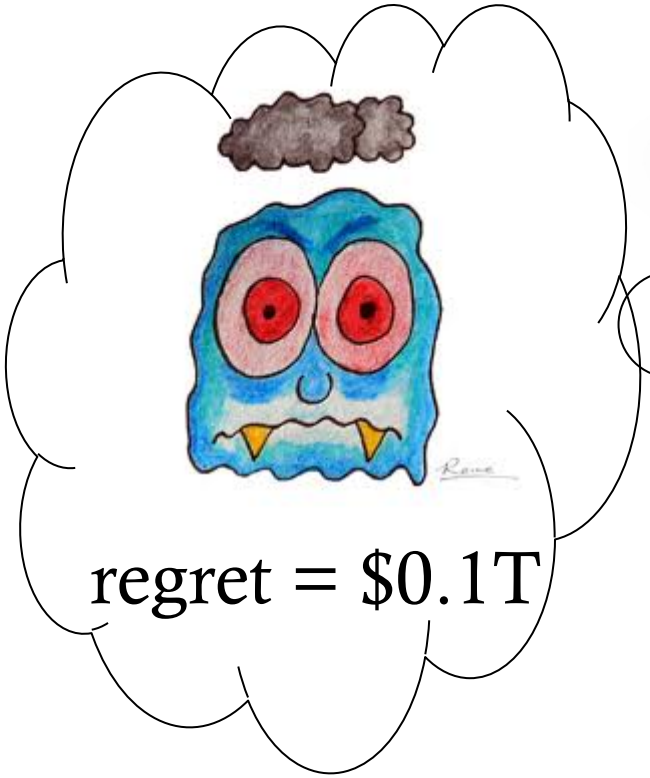
1  → **$0.5T**

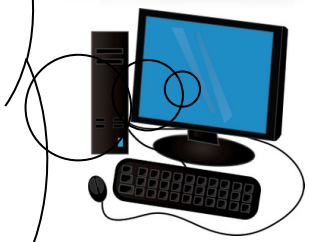
2  → $0.2T

3  → $0.33T

⋮

k  → $0.1T

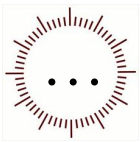


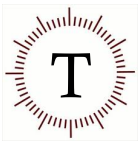


Contextual Bandits

[Auer-CesaBianchi-Freund-Schapire '02]

context:





1

2

3

⋮

k

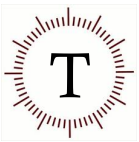
N experts/policies/functions

think of N >> K

Contextual Bandits

[Auer-CesaBianchi-Freund-Schapire '02]

context: x_1



⋮


5

1


1

4


K

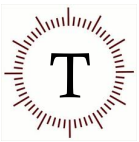
3

N experts/policies/functions
think of N >> K

Contextual Bandits

[Auer-CesaBianchi-Freund-Schapire '02]

context: x_1



1 

$0.15

2 

3 

⋮

k 

 5  1

 1  4

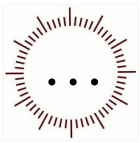
 K  3

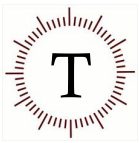
N experts/policies/functions
think of N >> K

Contextual Bandits

[Auer-CesaBianchi-Freund-Schapire '02]

context: x_1 x_2 x_3 x_4 … x_T





1  $0.15 $0.4

2 

3  $0.5 $0.2

⋮

k  $0.1

 $.12T  $.22T

 $.1T  0

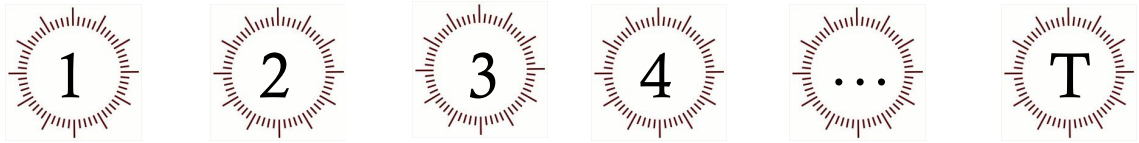
 $.2T  $.17T

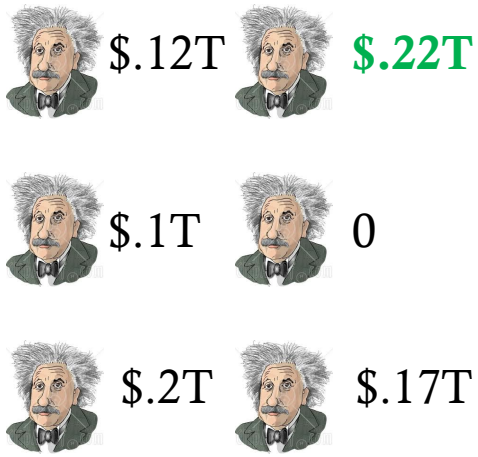
N experts/policies/functions
think of N >> K

Contextual Bandits

[Auer-CesaBianchi-Freund-Schapire '02]

context: x_1 x_2 x_3 x_4 … x_T



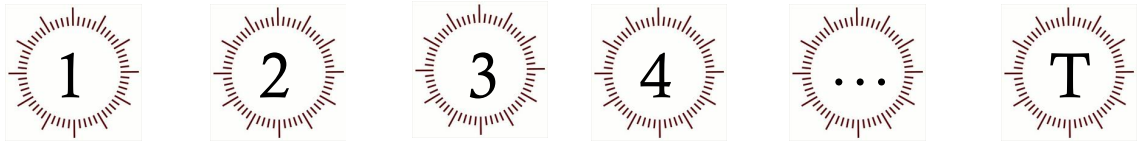


N experts/policies/functions
think of N >> K

Contextual Bandits

[Auer-CesaBianchi-Freund-Schapire '02]

context: x_1 x_2 x_3 x_4 … x_T

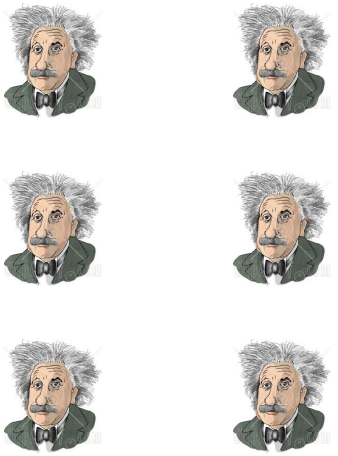


the rewards can come i.i.d. from a distribution or be arbitrary

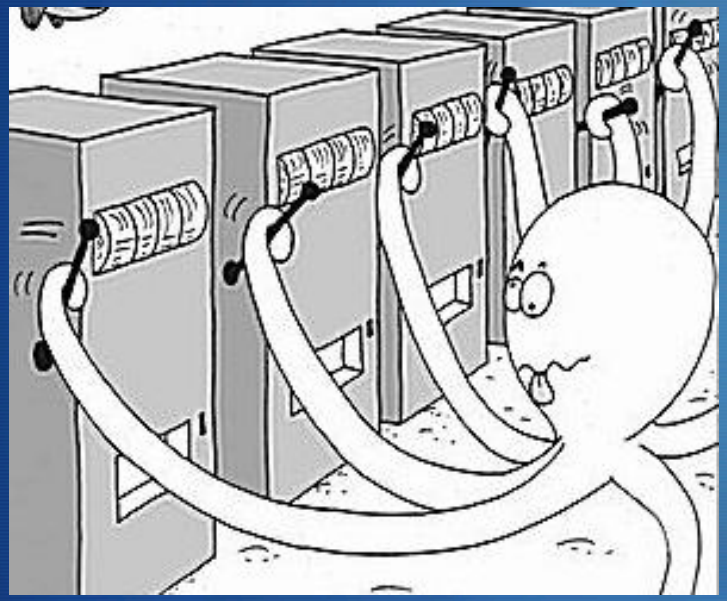
stochastic / adversarial

The experts can be present or not.

contextual / non-contextual



- Interaction
  - The feedback learner receives depends on its choice of arm.
  - Learner doesn't learn about arms/experts it didn't choose.
- Applications: "Life is a Bandit Problem"
  - Ad auctions: arms are advertisements, rewards are clicks, context is user data, policies are classifiers, one time-step is a user visit.
  - Medicine: arms are treatments, rewards are health outcomes, context is symptoms, policies are treatment plans, time-step is a patient visit.
  - Finance: arms are stocks, rewards are money earned, context is any news, policies are given by "financial experts," time-step is, say, a day.
- Exploration/Exploitation
  - Can exploit expert/arm you've learned to be good.
  - Can explore expert/arm you're not sure about.



The First Optimal Contextual Algorithm

[Beygelzimer-Langford-Li-**R**-Schapire '11]

Setting: Adversarial contextual bandits.

Result : The first optimal algorithm that works with high-probability. Also can have low regret w.r.t. a VC set.

Made Efficient in the Stochastic Case!

[Dudik-Hsu-Kale-Karampatziakis-Langford-**R**-Zhang '11]

Setting: Stochastic contextual bandits.

Result : The first efficient optimal, high probability algorithm (assuming a supervised-learning oracle)!

Some Barriers

$\Omega(kT)^{1/2}$ (non-contextual) and ~ $\Omega(TK \ln N)^{1/2}$ (contextual) are known lower bounds [Auer et al. '02] on regret, even in the stochastic case.

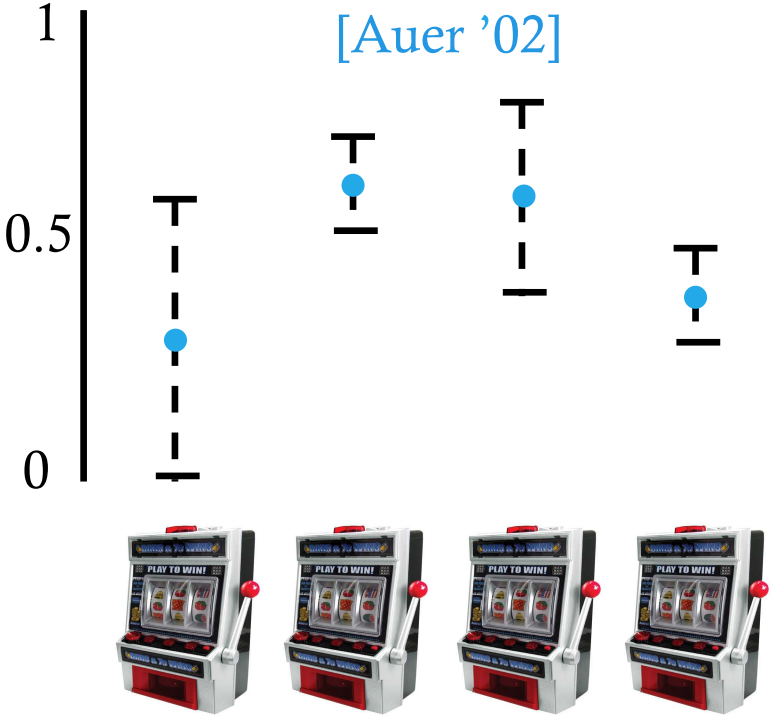
Any algorithm achieving regret $\tilde{O}(KT \text{ polylog } N)^{1/2}$ is said to be optimal.

No algorithm that first explores (acts randomly) and then exploits (follows the best policy) can be optimal. Any optimal algorithm must be adaptive.

Two Types of Approaches

UCB

[Auer '02]



Algorithm: at every time step

1) pull arm with highest UCB
2) update confidence bound of the arm pulled.

EXP3 Exponential Weights

[Littlestone-Warmuth '94]
[Auer et al. '02]

Algorithm: at every time step

1) sample from distribution defined by weights (mixed w/ uniform)
2) update weights "exponentially"

UCB vs EXP3
A Comparison

UCB
[Auer '02]

- Pros
  - Optimal for the stochastic setting.
  - Succeeds with high probability.
- Cons
  - Does not work in the adversarial setting.
  - Is not optimal in the contextual setting.

EXP3 & Friends
[Auer-CesaBianchi-Freund-Schapire '02]

- Pros
  - Optimal for both the adversarial and stochastic settings.
  - Can be made to work in the contextual setting
- Cons
  - Does not succeed with high probability in the contextual setting (only in expectation).

EXP4P
[Beygelzimer-Langford-Li-**R**-Schapire '11]

Main Theorem [Beygelzimer-Langford-Li-**R**-Schapire '11]: For any $\delta > 0$, with probability at least $1-\delta$, EXP4P has regret at most $O(KT \ln (N/\delta))^{1/2}$ in the adversarial contextual bandit setting.

EXP4P combines the advantages of Exponential Weights and UCB.

optimal for both the stochastic and adversarial settings

works for the contextual case (and also the non-contextual case)

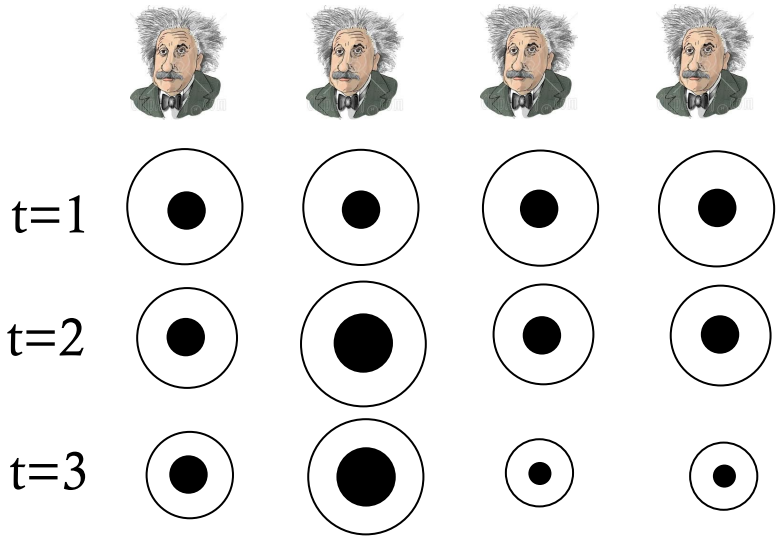
a high probability result

EXP4P
[Beygelzimer-Langford-Li-**R**-Schapire '11]

Main Theorem [Beygelzimer-Langford-Li-**R**-Schapire '11]: For any $\delta > 0$, with probability at least $1-\delta$, EXP4P has regret at most $O(KT \ln (N/\delta))^{1/2}$ in the adversarial contextual bandit setting.

key insights (on top of UCB/ EXP)

1) exponential weights and upper confidence bounds "stack"
2) generalized Bernstein's inequality for martingales



Hope for an Efficient Algorithm?

[Dudik-Hsu-Kale-Karampatziakis-Langford-**R**-Zhang '11]

For EXP4P, the dependence on N in the regret is logarithmic.

this suggests

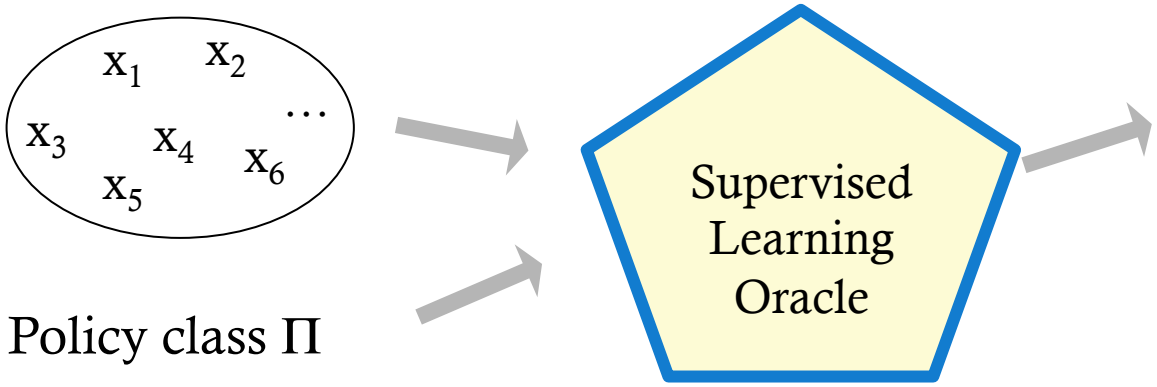
We could compete with a large, even super-polynomial number of policies! (e.g. $N=K^{100}$ becomes $10 \log^{1/2} K$ in the regret)

however

All known contextual bandit algorithms explicitly "keep track" of the N policies. Even worse, just reading in the N would take too long for large N.

Idea: Use Supervised Learning

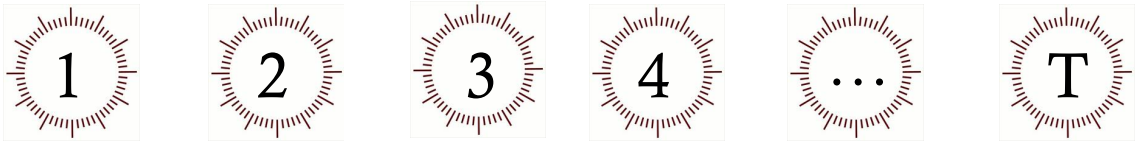
- "Competing" with a large (even exponentially large) set of policies is commonplace in supervised learning.
  - Targets: e.g. linear thresholds, CNF, decision trees (in practice only)
  - Methods: e.g. boosting, SVM, neural networks, gradient descent
- The recommendations of the policies don't need to be explicitly read in when the policy class has structure!



idea originates with [Langford-Zhang '07]

Back to Contextual Bandits

context: x_1 x_2 x_3



1 

$0.70

2 

$0.50

3 

⋮

k 

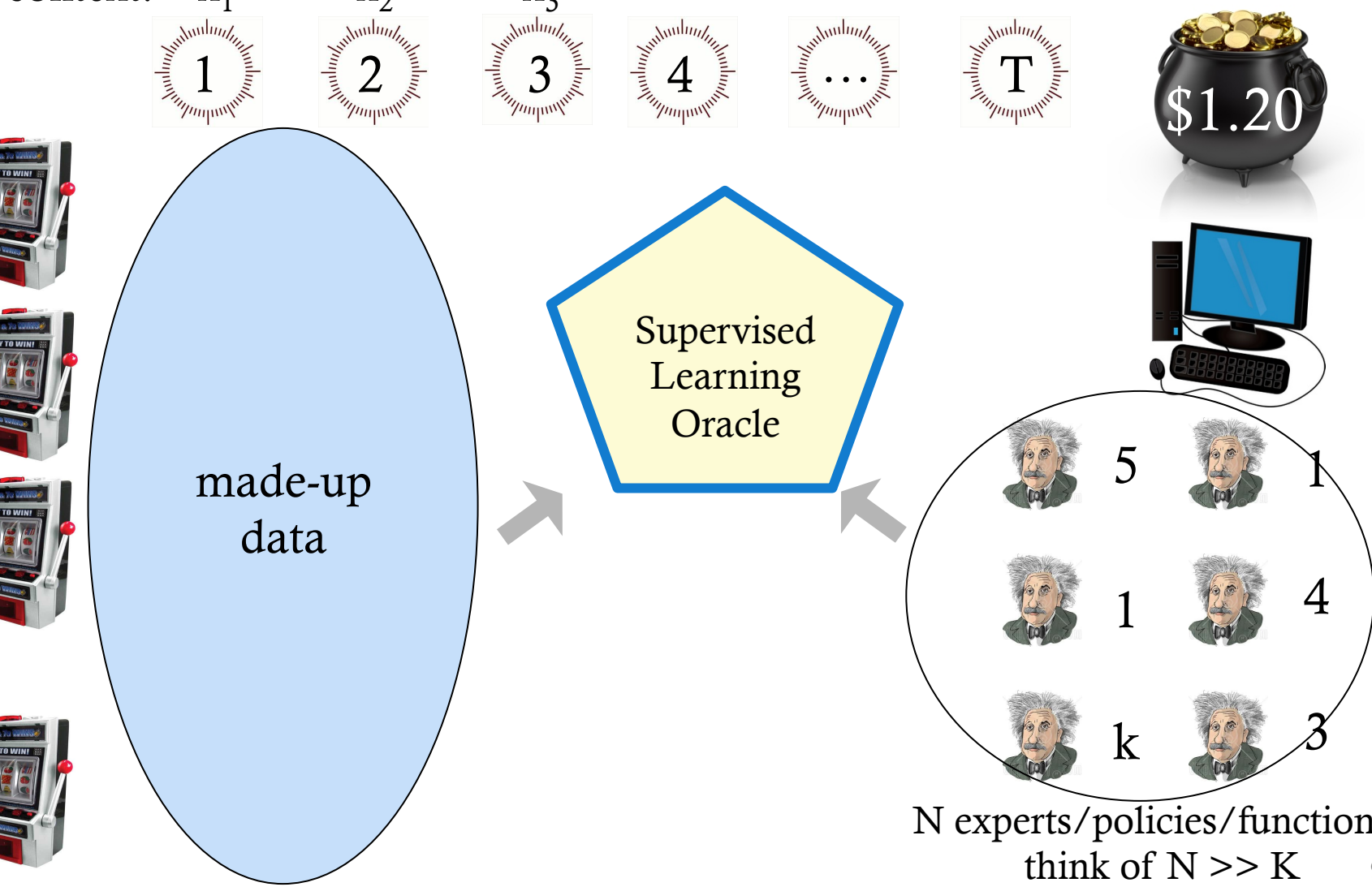

5 
1


1 
4

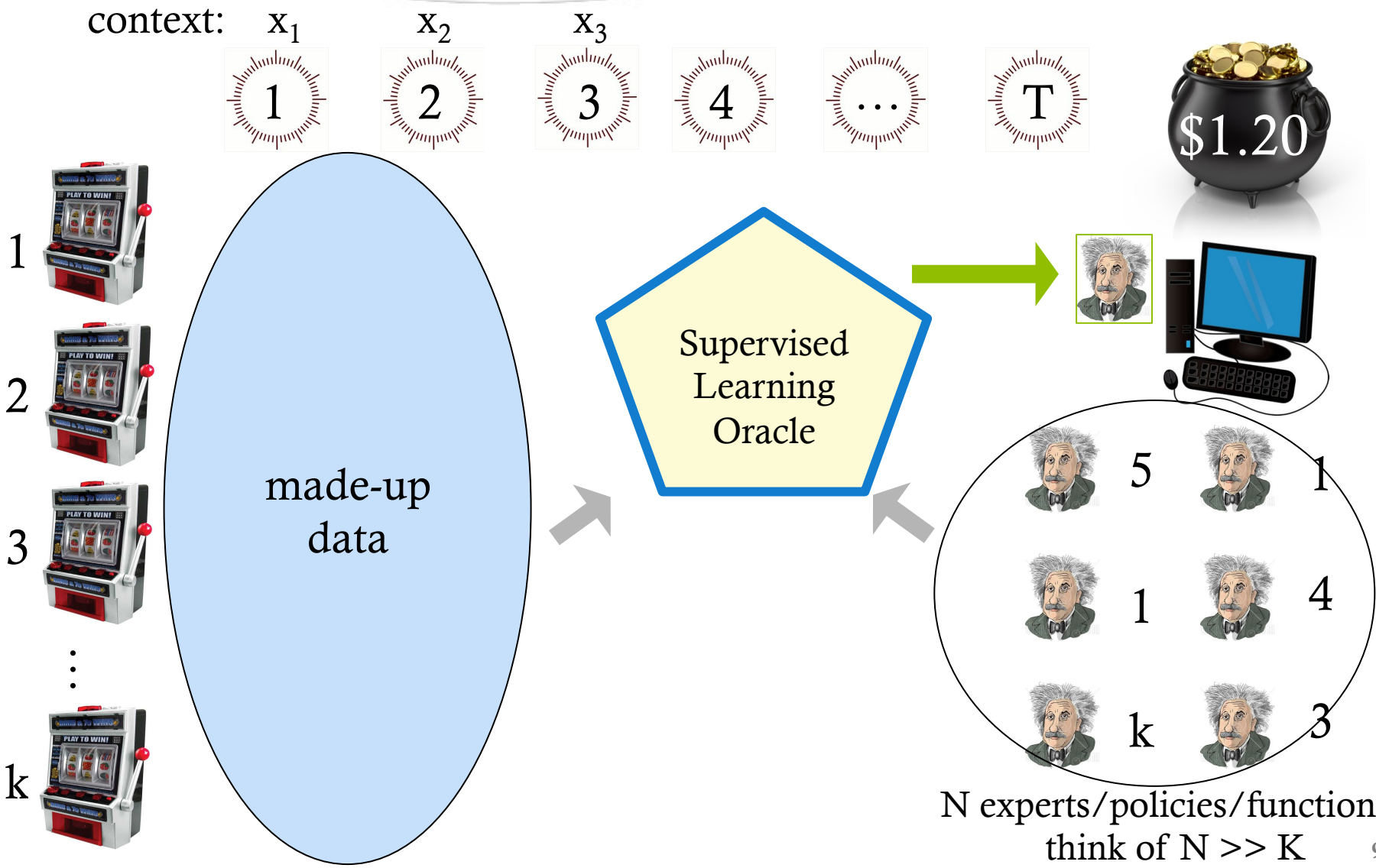

k 
3

N experts/policies/functions
think of N >> K

Back to Contextual Bandits



Back to Contextual Bandits



Randomized-UCB

Main Theorem [Dudik-Hsu-Kale-Karampatziakis-Langford-**R**-Zhang '11]: For any $\delta > 0$, w.p. at least $1-\delta$, given access to a supervised learning oracle, Randomized-UCB has regret at most $O((KT \ln(NT/\delta))^{1/2} + K \ln(NK/\delta))$ in the stochastic contextual bandit setting and runs in time $\text{poly}(K, T, \ln N)$.

Randomized-UCB

Main Theorem [Dudik-Hsu-Kale-Karampatziakis-Langford-**R**-Zhang '11]: For any $\delta > 0$, w.p. at least $1-\delta$, given access to a supervised learning oracle, Randomized-UCB has regret at most $O((KT \ln(NT/\delta))^{1/2} + K \ln(NK/\delta))$ in the stochastic contextual bandit setting and runs in time $\text{poly}(K, T, \ln N)$.

if arms are chosen among only good policies s.t. all have variance < approx 2K, we win

can prove this exists via a minimax theorem

this condition can be softened to occasionally allow choosing of bad policies

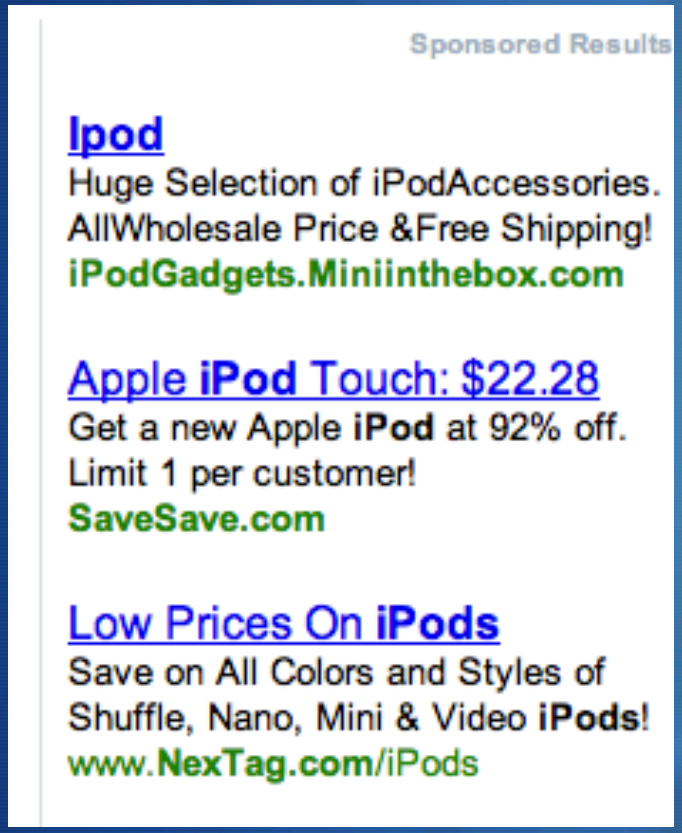
via "randomized" upper confidence bounds

creates an optimization problem of how to choose arms

expressed as convex program

solvable by ellipsoid algorithm

can implement a separation oracle with the supervised learning oracle



Bandit Slate Problems

[Kale-**R**-Schapire '11]

Problem: Instead of selecting one arm, we need to select $s \geq 1$, arms (possibly ranked). The motivation is web ads where a search engine shows multiple ads at once.

Result: Efficient algorithm with regret $\tilde{O}(sKT)^{1/2}$ for unranked slates and $\tilde{O}(s(KT)^{1/2})$ for ranked slates.

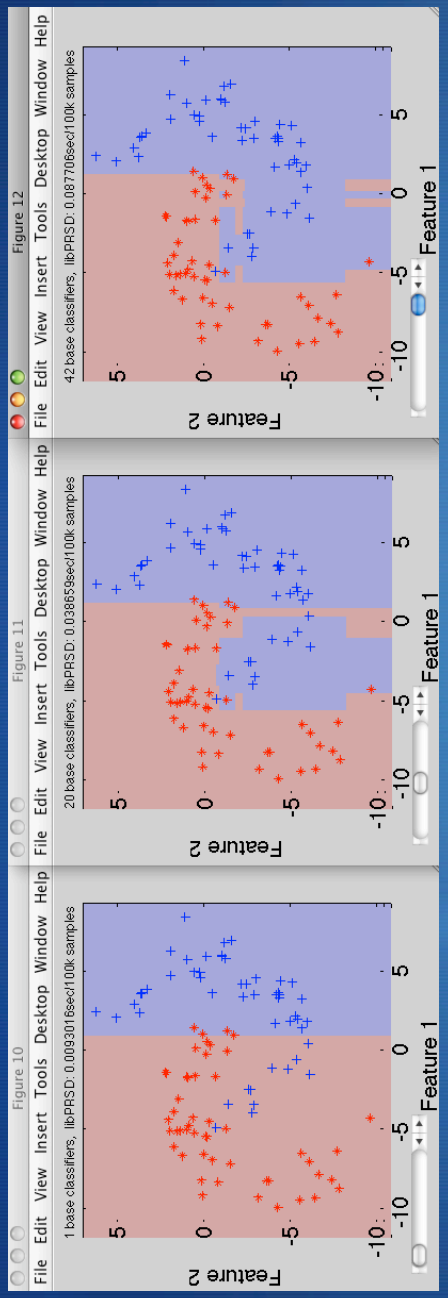
Linear Bandits

[Chu-Li-**R**-Schapire '11]

Problem: The LinUCB algorithm [Li-Chu-Langford-Schapire '10] was designed to compete with any linear predictor. It was experimentally shown to be effective but had no theoretical analysis.

Result: We proved that a variant LinUCB was optimal for the linear setting and by giving an analysis and an almost matching lower bound. Afterwards, a direct proof for the regret of LinUCB was found [AbbasiYadkori-Pal- Szepesvári '11]

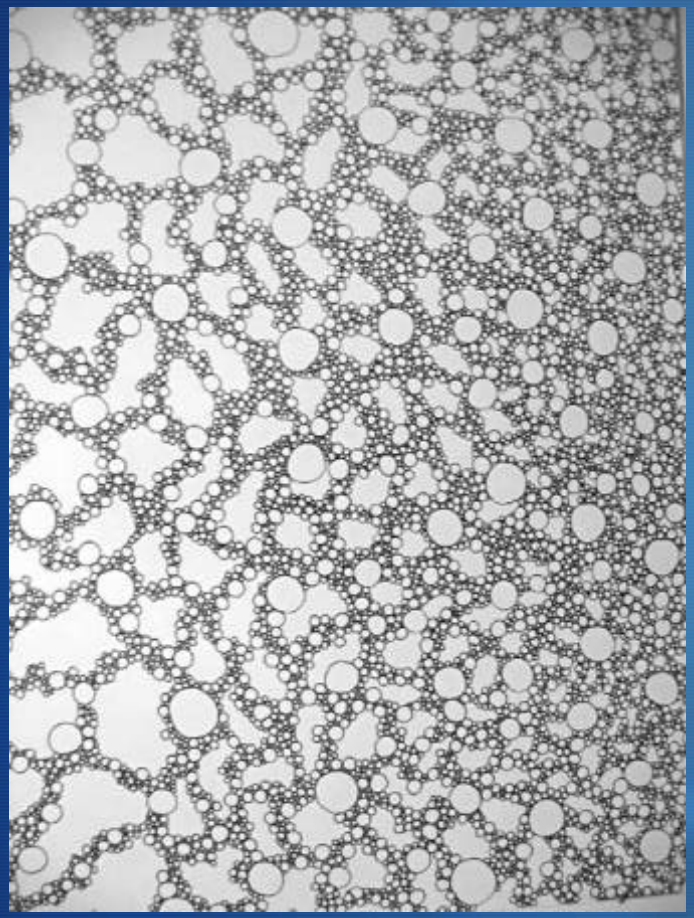
Other Research



Margins in Boosting
[R-Schapire '06]

Setting: The algorithm AdaBoost [Freund-Schapire '97] combines the votes of "weak learners" to produce a "strong learner." It was a mystery why AdaBoost does not overfit. The margins theory [Schapire et al. '98] explained this mystery, but was put into doubt by arc-gv [Breiman '99].

Result: We showed that results on arc-gv actually don't contradict the margins theory and discovered a complex interplay between margins, weak learner complexity, and training sample size.

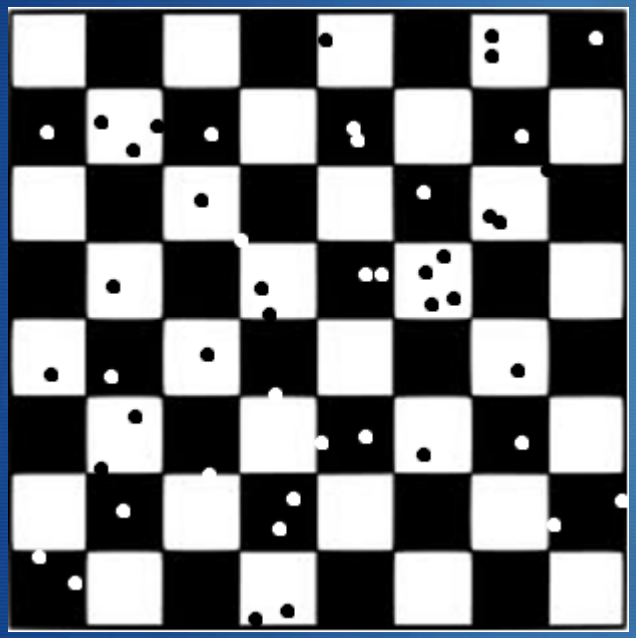


Embedding Parity into Random Structures

[Angluin-Eisenstat-Kontorovich-**R** '10]

Setting: Cryptographic hardness [Kearns-Valiant '89] results prevent progress for learning automata. Hence we considered the case of learning **random automata**.

Result: Even random automata run into a barrier: the parity function can be embedded into a random automata (and even decision trees and DNF) w.h.p. Hence, no **statistical query algorithm** [Kearns '98] can hope to learn these targets.



Improved Bounds for Noisy Sparse Parity

[Grigorescu-**R**-Vempala '11]

Setting: The parity problem in the presence of noise is a difficult problem on the forefront of theoretical computer science. No algorithm better than the brute force $O(n^r)$ for r-parities.

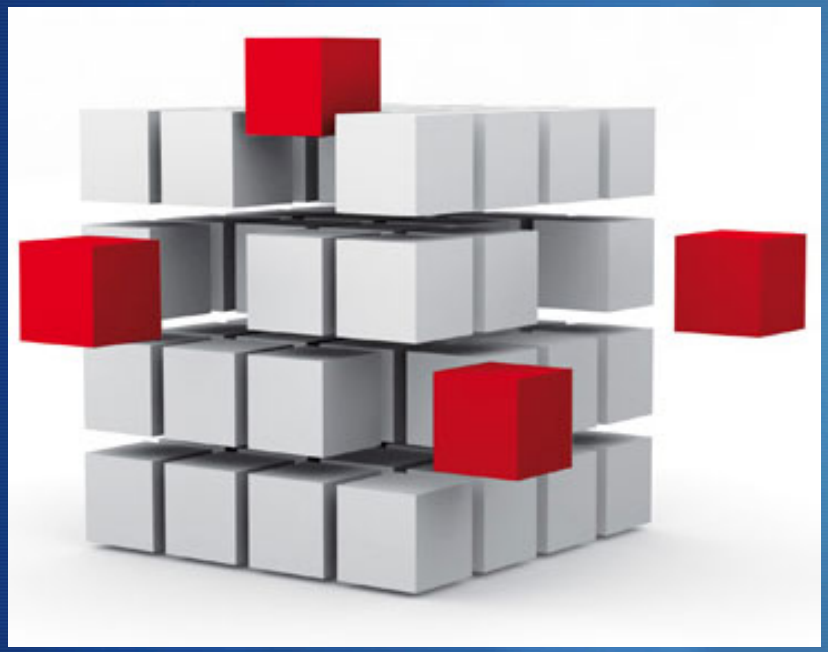
Result: We gave the first nontrivial algorithm for this problem, giving a $\sim O(n^{r/2})$ bound. This also implies slightly better bounds for noisy r-juntas and noiseless s-term DNF under the uniform distribution.

Complexity of Statistical Algorithms

[Feldman-Grigorescu-**R**-Vempala '11]

Setting: For optimization problems over distributions, most current algorithms, e.g. local search, MCMC, gradient descent, etc. can be seen as "statistical."

Result: We define a parameter that gives unconditional lower bounds on the number of samples a statistical algorithm requires, indicating limitations of known heuristics for many problems. This generalizes the statistical query work of [Blum et al '94].



Feature-Efficient Prediction

[R '11]

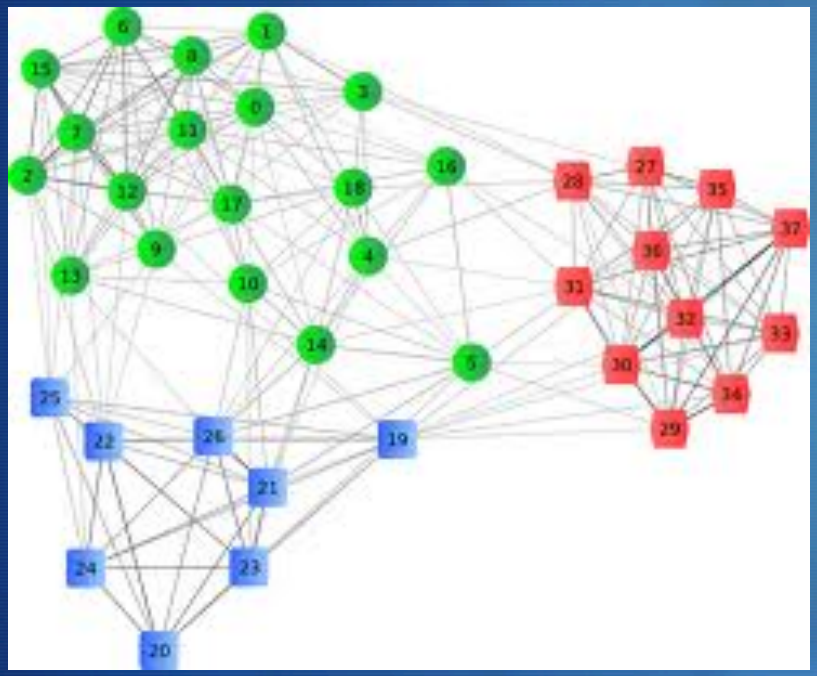
Setting: The learner has a budget of how many features it can examine at test-time.

Application: Medical testing, where in diagnosing a patient, tests can be harmful and/or expensive.

Result: Properly sampling from any ensemble of feature-efficient predictors allows the learner's error bound to converge to the margin bound of a full ensemble.

Stability Assumptions in Clustering

[R '11]



Setting: Clustering has recently been studied [Awasthi-Blum-Sheffet '11] under the perturbation resilience assumption [Bilu-Linial '10] -- that small changes to pair-wise distances ought not affect the optimal clustering.

Result: Our analysis reveals that for the k-median and min-sum objectives, for only a small range of stability parameters is the problem interesting. Outside that range, the clustering problem either remains NP-hard or becomes trivial.

Future Work

Learning Networks

- My research has addressed problems in evolutionary tree reconstruction, gene sequencing, protein networks, social networks, and language learning.
- Has immediate applications to learning computer networks, chemical networks, spreads of disease, etc.
  - Can we find more applications?
  - New applications will perhaps lead to new models and new problems!
  - Some general questions remain open: for example, taking query costs into account.

Bandit Problems

- "Life is a bandit problem" – so we have many applications!
- Open problem: making the reduction to supervised learning more practical and able to handle the adversarial case.
- Open problem: develop more general theory for the reinforcement learning problem.
  - Multiarmed bandits are just a special case!
- More experiments!

Other Directions
a small sampling

- Fast prediction with strong guarantees.
  - Can we capture prediction accuracy and prediction time in one optimization?
- Understanding the true complexity of statistical algorithms.
- Better models for learning social interactions / networks.
  - What are the most robust (ie. to disease) networks?
  - What networks are we currently building?
- Many others…

Thank You!

Questions?