Restriction Access, Population Recovery & Partial Identification

Avi Wigderson
IAS, Princeton

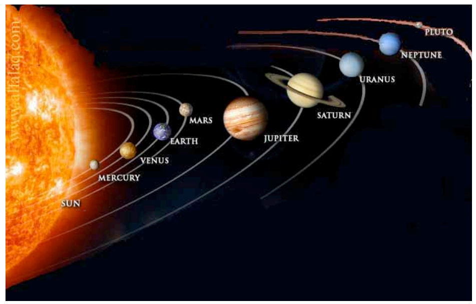
Joint with

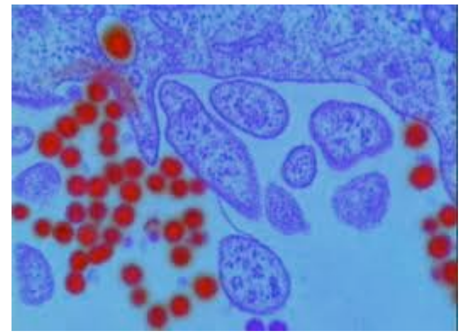
Zeev Dvir
Anup Rao
Amir Yehudayoff

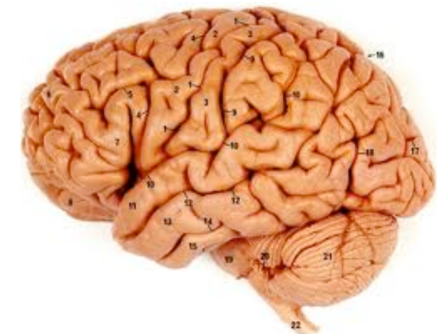
Restriction Access,

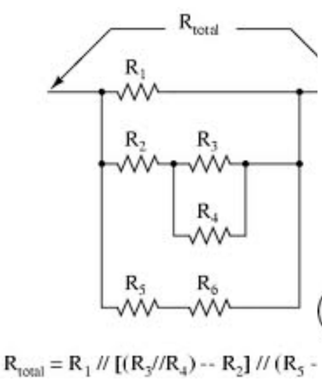
A new model of “Grey-box” access

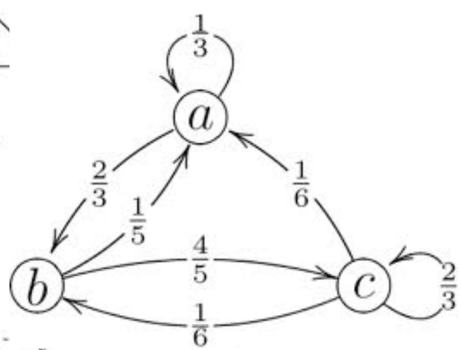
Systems, Models, Observations

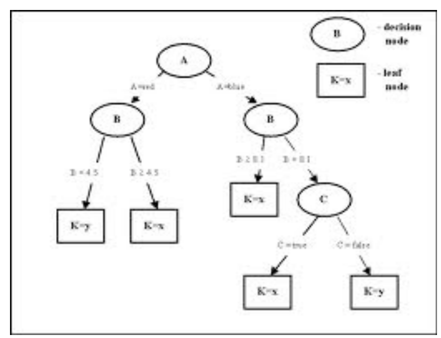


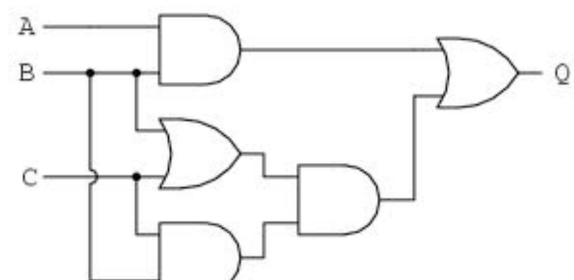












From Input-Output $(I_1,O_1), (I_2,O_2), (I_3,O_3)$, ….?

Typically more!

Black-box access

Successes & Limits

Learning: PAC, membership, statistical…queries

Decision trees, DNFs?

Cryptography: semantic, CPA, CCA, … security

Cold boot, microwave,… attacks?

Optimization: Membership, separation,… oracles

Strongly polynomial algorithms?

Pseudorandomness: Hardness vs. Randomness

Derandomizing specific algorithms?

Complexity: $\Sigma^2 = NP^{NP}$

What problems can we solve if P=NP?

The gray scale of access

$f: \Sigma^n \to \Sigma^m$

D: "device" computing f

(from a family of devices)

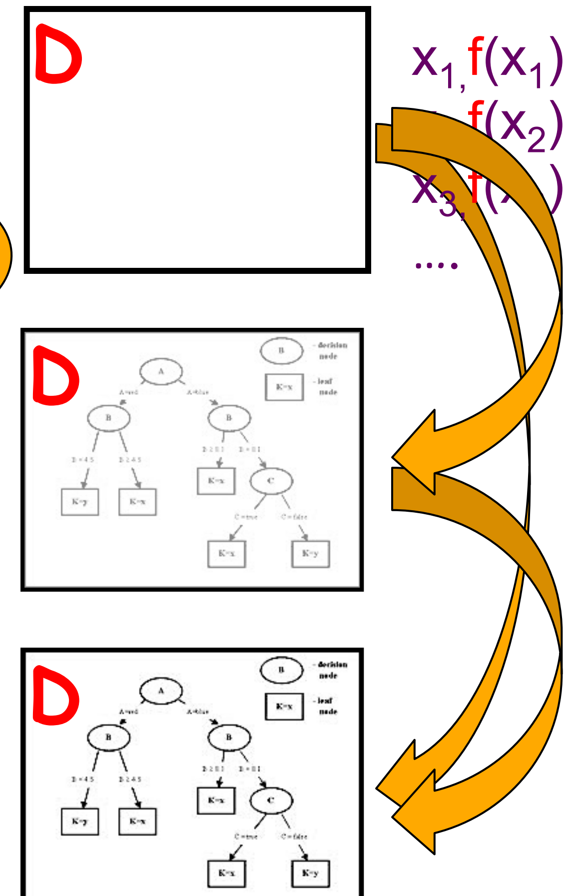
Black Box

How to model?
Many specific ideas.
Ours: general, clean

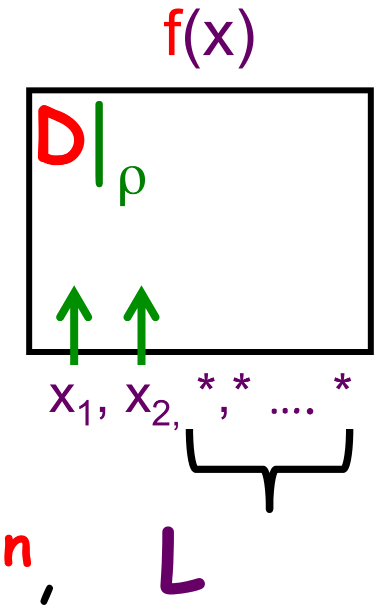
Gray Box

- natural starting point
- natural intermediate pt

Clear Box



Restriction Access (RA)



$f: \Sigma^n \rightarrow \Sigma^m$

D: "device" computing f

Restriction: $\rho = (x, L)$, $L \subseteq [n]$, $x \in \Sigma^n$,

L live vars

Observations: $(\rho, D|_\rho)$

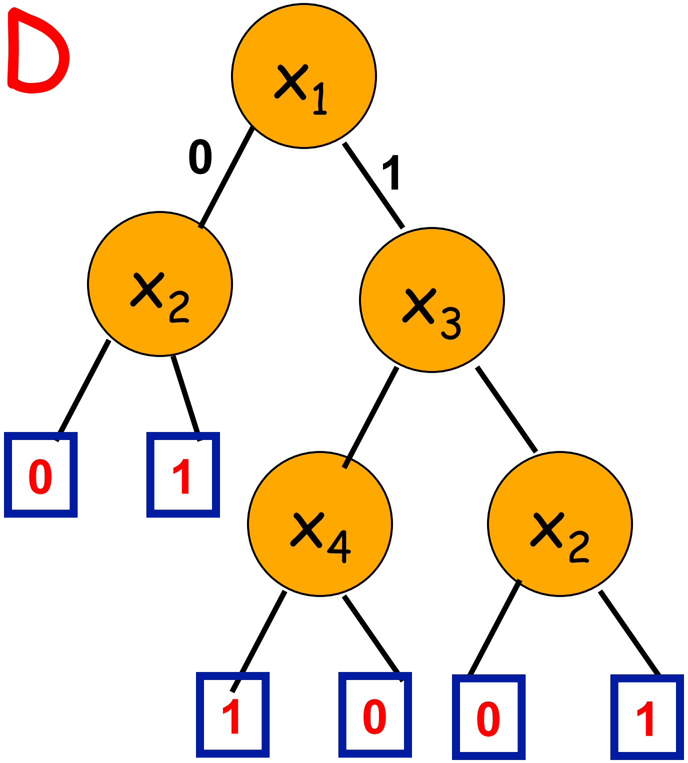
$D|_\rho$ (simplified after fixing) computes $f|_\rho$ on L

Black $L = \phi$ Gray Clear $L = [n]$

$(x, f(x))$ $(\rho, D|_\rho)$ (x, D)

Example: Decision Tree

D

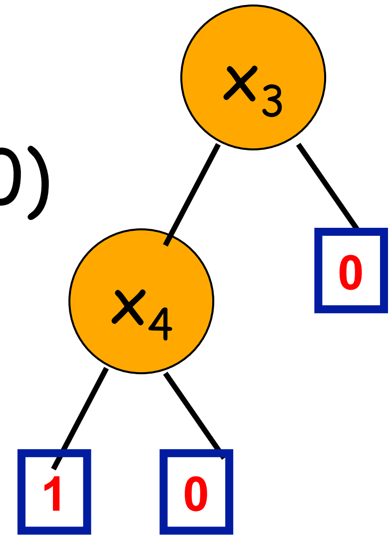


$\rho = (x, L)$

$L = \{3,4\}$

$x = (1010)$

$D|_\rho =$



Modeling choices (RA-PAC)

Restriction: $\rho = (x,L)$, $L \subseteq [n]$, $x \in \Sigma^n$, unknown D

Input x : friendly, adversarial, random

Unknown distribution (as in PAC)

Live vars L : friendly, adversarial, random

μ-independent dist (as in random restrictions)

RA-PAC Results

Positive - In contrast to PAC !!!

Probably, Approximately Correct (PAC) learning of D, from restrictions with each variable remains alive with prob μ

Thm 1[DRWY]: A poly(s, μ) alg for RA-PAC learning size-s **decision trees**, for every $\mu>0$

(reconstruction from pairs of live variables)

Thm 2[DRWY]: A poly(s, μ) alg for RA-PAC learning size-s **DNFs**, for every $\mu > .365\ldots$

(reduction to "Population Recovery Problem")

Population Recovery

(learning a mixture of binomials)

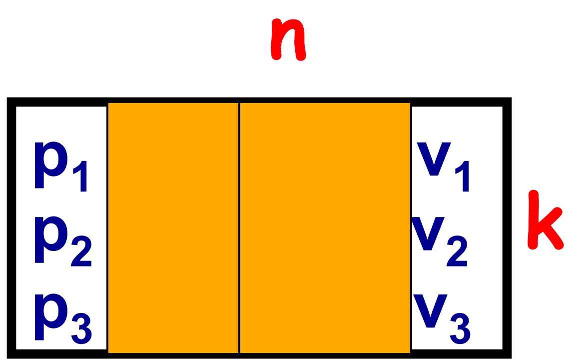
Population Recovery Problem

k species, n attributes, from Σ,

Vectors $v_1, v_2, \ldots v_k \in \Sigma^n$

Distribution $p_1, p_2, \ldots p_k$

μ, ε >0



Red: Known

Blue: Unknown

Task: Recover all v_i, p_i (upto ε) from samples

Population Recovery Problem

k species, n attributes, from Σ, $\mu, \varepsilon > 0$

$v_1, v_2, \ldots v_k \in \Sigma^n$

$p_1, p_2, \ldots p_k$ fraction in population

p_1	1/2	0000	v_1
p_2	1/3	0110	v_2
p_3	1/6	1100	v_3

Task: Recover all v_i, p_i (upto ε) from samples

Samplers:

(1) $u \leftarrow v_i$ with prob. p_i

0110

μ-Lossy Sampler:

(2) $u(j) \leftarrow ?$ with prob. $1-\mu$ $\forall j \in [n]$

?1?0

μ-Noisy Sampler:

(2) $u(j)$ flipped w.p. $1/2-\mu$ $\forall j \in [n]$

1100

Loss - Paleontology

Skull Teeth Vertebrae Arms Ribs Legs Tail

26% 

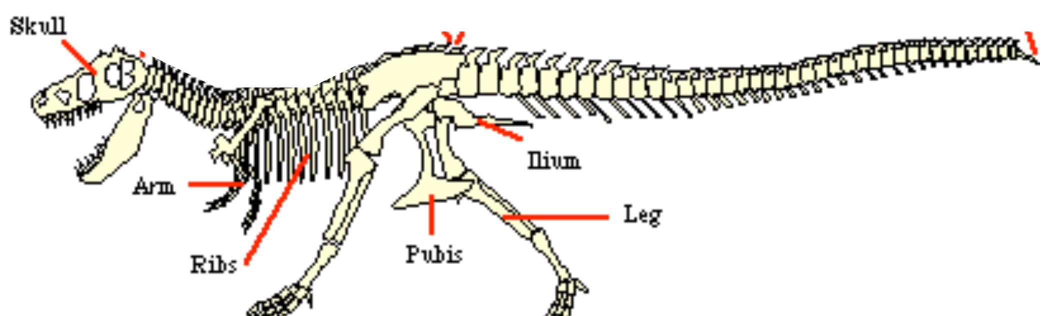
11% 

True Data

13% 

30% 

20% 



Loss - Paleontology

From samples

Skull Teeth Vertebrae Arms Ribs Legs Tail

Dig #1

Dig #2

Dig #3

Dig #4 …… each finding common to many species!

How do they do it?

Noise - Privacy

	Socialism	Abortion	Gay marriage	Marijuana	Male	Rich	North US
2%	0	1	1	0	1	0	0
1%	1	1	0	0	0	1	1
……	……						

True Data

From samples

Joe	0	0	0	0	0	1	1
Jane	0	0	0	0	1	1	1

….Who flipped every correct answer with probability 49%

Deniability? Recovery?

PRP - applications

Recovering from loss & noise

- Clustering / Learning / Data mining
- Computational biology / Archeology / ......
- Error correction
- Database privacy
- ......

Numerous related papers & books

PRP - Results

Facts: μ=0 obliterates all information.

- No polytime algorithm for $\mu = o(1)$

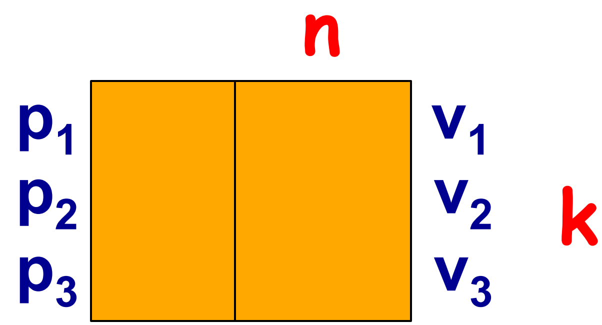
Thm 3 [DRWY] A poly(k, n, ε) algorithm, from **lossy** samples, for every $\mu > .365...$

Thm 4 [WY]: A poly($k^{\log k}$, n, ε) algorithm, from **lossy** and/or **noisy** samples, for every $\mu > 0$

Kearns, Mansour, Ron, Rubinfeld, Schapire, Sellie
exp(k) algorithm for this discrete version
Moitra, Valiant
exp(k) algorithm for Gaussian version
(even when noise is unknown)

Proof of Thm 4

Reconstruct v_i, p_i



From samples ?1?0, 0??0, 1100,....

Lemma 1: Can assume we know the v_i's !

Proof: Exposing one column at a time. ■

Lemma 2: Easy in exp(n) time !

Proof: Lossy - enough samples without "?"

Noisy – linear algebra on sample probabilities.

Idea: Make n=O(log k) [Dimension Reduction]

Partial IDs

a new dimension-reduction technique

Dimension Reduction and small IDs

$n = 8$
$k = 9$

	1	2	3	4	5	6	7	8	
p_1	0	0	0	0	0	1	0	1	v_1
p_2	0	1	1	0	1	0	1	0	v_2
p_3	0	1	0	0	1	0	1	1	v_3
p_4	1	1	1	0	1	0	1	1	v_4
p_5	1	1	0	0	0	1	1	1	v_5
p_6	1	1	0	0	1	0	0	1	v_6
p_7	0	1	0	0	0	1	1	1	v_7
p_8	1	1	0	1	1	0	1	1	v_8
p_9	1	1	0	0	0	1	1	1	v_9

IDs

$S_1 = \{1,2\}$
$S_2 = \{8\}$
$S_3 = \{1,5,6\}$

u – random sample

$q_i = \Pr[u[S_i]=v_i[S_i]]$

Lemma: Can approximate p_i in $\exp(|S_i|)$ time !

Does one always have small IDs?

Small IDs ?

$n = 8$
$k = 9$

	1	2	3	4	5	6	7	8		IDs
p_1	1	0	0	0	0	0	0	0	v_1	$S_1 = \{1\}$
p_2	0	1	0	0	0	0	0	0	v_2	$S_2 = \{2\}$
p_3	0	0	1	0	0	0	0	0	v_3	$S_3 = \{3\}$
p_4	0	0	0	1	0	0	0	0	v_4	…
p_5	0	0	0	0	1	0	0	0	v_5	
p_6	0	0	0	0	0	1	0	0	v_6	
p_7	0	0	0	0	0	0	1	0	v_7	
p_8	0	0	0	0	0	0	0	1	v_8	$S_8 = \{8\}$
p_9	0	0	0	0	0	0	0	0	v_9	$S_9 = \{1,2,\dots,8\}$

NO!

However,…

Linear algebra & Partial IDs

$n = 8$
$k = 9$

	1	2	3	4	5	6	7	8		PIDs
p_1	1	0	0	0	0	0	0	0	v_1	$S_1 = \{1\}$
p_2	0	1	0	0	0	0	0	0	v_2	$S_2 = \{2\}$
p_3	0	0	1	0	0	0	0	0	v_3	$S_3 = \{3\}$
p_4	0	0	0	1	0	0	0	0	v_4	…
p_5	0	0	0	0	1	0	0	0	v_5	
p_6	0	0	0	0	0	1	0	0	v_6	
p_7	0	0	0	0	0	0	1	0	v_7	
p_8	0	0	0	0	0	0	0	1	v_8	$S_8 = \{8\}$
p_9	0	0	0	0	0	0	0	0	v_9	$S_9 = \emptyset$

However, we can compute $p_9 = 1 - p_1 - p_2 - \ldots - p_8$

Back substitution and Imposters

		1	2	3	4	5	6	7	8	
$q_1 =$	p_1	0	0	1	0	0	1	0	1	v_1
$q_2 =$	p_2	0	1	1	0	1	0	1	0	v_2
$q_3 =$	p_3	0	1	0	0	1	0	1	1	v_3
$q_4 - p_1 - p_2 =$	p_4	1	1	1	0	1	0	1	1	v_4
	p_5	1	1	0	0	0	1	1	1	v_5
	p_6	1	1	0	0	1	0	0	1	v_6
	p_7	0	1	0	0	0	1	1	1	v_7
	p_8	1	1	0	1	1	0	1	1	v_8
	p_9	1	1	0	0	0	1	1	1	v_9

PIDs

$S_1 = \{1,2\}$

$S_2 = \{8\}$

$S_3 = \{1,5,6\}$

$S_4 = \{3\}$

any subset

u - random sample

$q_i = \Pr[u[S_i]=v_i[S_i]]$

Can use back substitution **if** no cycles !

Are there always acyclic small ***partial*** IDs?

Acyclic small *partial* IDs exist

$n = 8$
$k = 9$

	1	2	3	4	5	6	7	8	
p_1	[redacted]								v_1
p_2									v_2
p_3	[redacted]								v_3
p_4									v_4
p_5									v_5
p_6									v_6
p_7									v_7
p_8	~~0~~	~~1~~	~~0~~	~~0~~	~~0~~	~~1~~	~~1~~	~~1~~	v_8
p_9	[redacted]								v_9

PIDs

$S_8 = \{1,5,6\}$

Lemma: There is always an ID of length log k

Idea: Remove and iterate to find more PIDs

Lemma: Acyclic (log k)-PIDs always exists!

Chains of small Partial IDs

$n = 8$
$k = 6$

	1	2	3	4	5	6	7	8		PIDs
p_1	1	1	1	1	1	1	1	1	v_1	$S_1 = \{1\}$
p_2	0	1	1	1	1	1	1	1	v_2	$S_2 = \{2\}$
p_3	0	0	1	1	1	1	1	1	v_3	$S_3 = \{3\}$
p_4	0	0	0	1	1	1	1	1	v_4	…
p_5	0	0	0	0	1	1	1	1	v_5	
p_6	0	0	0	0	0	1	1	1	v_6	$S_6 = \{6\}$

Compute: $q_i = \Pr[u_i = 1] = \Sigma_{j \le i}\, p_i$ from sample u

Back substitution: $p_i = q_i - \Sigma_{j<i}\, p_j$

Problem: Long chains! Error doubles each step, so is exponential in the chain length.

Want: Short chains!

The PID (imposter) graph

Given: $V=(v_1, v_2, \dots v_k) \in \Sigma^n$ $S=(S_1, S_2, \dots, S_k) \subseteq [n]^n$

Construct $G(V;S)$ by connecting $v_j \rightarrow v_i$ **iff**

v_i **is an imposter of** v_j : $v_i[S_j] = v_j[S_j]$

1	2	3	4	5	6	7	8	
1	1	1	1	1	1	1	1	v_1
0	1	1	1	1	1	1	1	v_2
0	0	1	1	1	1	1	1	v_3
0	0	0	1	1	1	1	1	v_4
0	0	0	0	1	1	1	1	v_5

$v_i \rightarrow v_j$ iff $i > j$

PIDs

$S_1 = \{1\}$

$S_2 = \{2\}$

$S_3 = \{3\}$

...

$S_5 = \{5\}$

width $= \max_i |S_i|$ **depth** $= \text{depth}(G)$

Want: PIDs w/small width and depth for all **V**

Constructing cheap PID graphs

Theorem: For every $V=(v_1, v_2, \ldots v_k)$, $v_i \in \Sigma^n$ we can efficiently find PIDs $S=(S_1,S_2,\ldots,S_k)$, $S_i \subseteq [n]$ of width and depth at most log k

Algorithm: **Initialize** $S_i=\varnothing$ for all **i**

Invariant: $|imposters(v_i;S_i)| \leq k/2^{|S_i|}$

Repeat: (1) Make S_i maximal

if not, add minority coordinates to S_i

(2) Make chains monotone:

$v_j \rightarrow v_i$ then $|S_j|<|S_i|$ (so G acyclic)

if not, set S_i to S_j (and apply (1) to S_i)

1	2	3	4	
0	0	1	0	v_1
0	0	0	0	v_2
0	0	0	1	v_3
1	0	0	1	v_4
1	1	1	0	v_5
1	0	1	0	v_6

Analysis of the algorithm

Theorem: For every $V=(v_1, v_2, \ldots v_k) \in \Sigma^n$ we can efficiently find PIDs $S=(S_1,S_2,\ldots,S_k) \subseteq [n]^n$ of width and depth at most log k

Algorithm: **Initialize** $S_i=\emptyset$ for all **i**

Invariant: $|\text{imposters}(v_i;S_i)| \le k/2^{|S_i|}$

Repeat: (1) Make S_i maximal

(2) Make chains monotone ($v_j \rightarrow v_i$ then $|S_j|<|S_i|$)

Analysis:

- $|S_i| \le \log k$ throughout for all **i**
- $\sum_i |S_i|$ increases each step
- Termination in $k \log k$ steps.
- **width** $\le \log k$ and so **depth** $\le \log k$

Conclusions

- Restriction access: a new, general model of "gray box" access (largely unexplored!)

- A general problem of population recovery

- Efficient reconstruction from loss & noise

- Partial IDs, a new dimension reduction technique for databases.

Open: polynomial time algorithm in k ?
(currently $k^{\log k}$, PIDs can't beat $k^{\log\log k}$)

Open: Handle unknown errors ?