Sub-Linear Time Algorithms: Fast, Cheap and (Only a Little) Out of Control

Ronitt Rubinfeld

MIT and Tel Aviv U.

Algorithms for REALLY big data

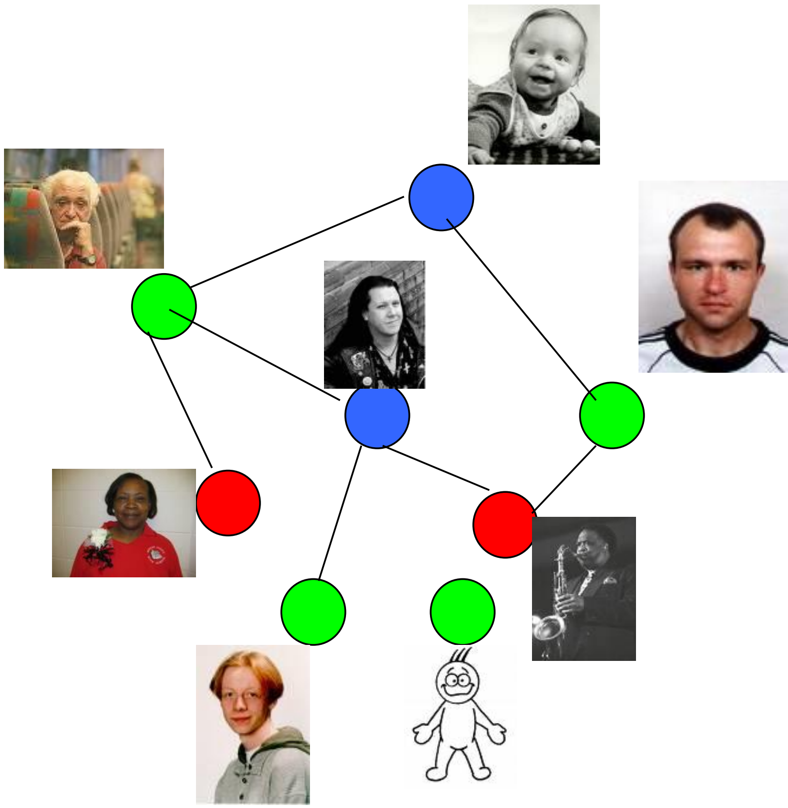
Part I

No time

What can we hope to do without viewing most of the data?

Small world phenomenon

- The social network is a graph:
  - "node" is a person
  - "edge" between people that know each other
- "6 degrees of separation"
  - Are all pairs of people connected by path of distance at most 6?



DOES IT HOLD?

Vast data

- Impossible to access all of it

- Accessible data is too enormous to be viewed by a single individual

- Once accessed, data can change

The Gold Standard

- linear time algorithms!
- Inadequate…

What can we hope to do without viewing most of the data?

- Can’t answer “for all” or “exactly” type statements:
  - exactly how many individuals on earth are left-handed?
  - are all individuals connected by at most 6 degrees of separation?

- Compromise?
  - approximately how many individuals on earth are left-handed?
  - is there a large group of individuals connected by at most 6 degrees of separation?

Types of approximation:

Property testing

Traditional approximation

“In the ballpark” vs. “out of the ballpark” tests

- Property testing: Distinguish inputs that have specific property from those that are *far* from having that property

- Benefits:
  - Can often answer such questions *much faster*
  - May be the natural question to ask
    - When some “noise” always present
    - When data constantly changing
    - Gives fast sanity check to rule out very “bad” inputs
    - Model selection problem in machine learning

Property testing

Requirements of property tester:

- if input has property, tester passes (whp)
- if input ϵ-far from all inputs with property, tester fails (whp)

("in between cases" – ok for tester to pass OR fail)

What is ϵ-far?

Need to specify –
usually close in Hamming distance

Sortedness of a sequence

- Given: list $y_1\, y_2 \ldots\, y_n$
- Question: is the list sorted?

- Clearly requires n steps – must look at each y_i

Sortedness of a sequence

- Given: list $y_1 \, y_2 \ldots \, y_n$

- Question: can we quickly test if the list close to sorted?

What do we mean by ``quick’’?

- query complexity measured in terms of list size *n*

- Our goal (if possible):
  - *Very small* compared to *n,* will go for *clog n*

What do we mean by "close''?

Definition: a list of size n is ε-close to sorted if can delete at most εn values to make it sorted. Otherwise, ε-far

(ε is given as input, e.g., ε=1/10)

Sorted: 1 2 4 5 7 11 14 19 20 21 23 38 39 45

Close: 1 4 2 5 7 11 14 19 20 39 23 21 38 45

1 4 5 7 11 14 19 20 23 38 45

Far: 45 39 23 1 38 4 5 21 20 19 2 7 11 14

1 4 5 7 11 14

Requirements for algorithm:

- Pass sorted lists
- Fail lists that are ε-far
  - Equivalently: if list likely to pass test, can change at most ε fraction of list to make it sorted

What if list not sorted, but not far?

Probability of success > ¾

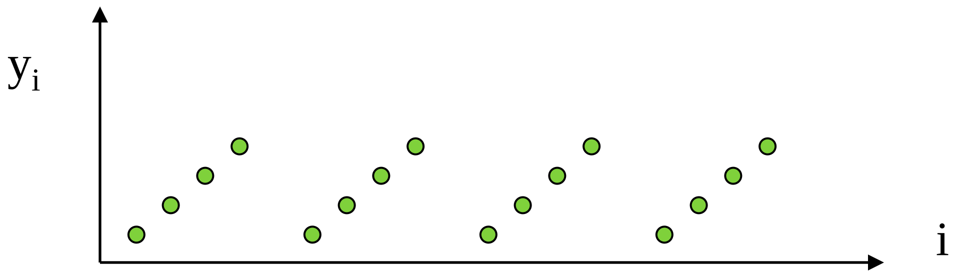
(can boost it arbitrarily high by repeating several times. Then output "fail" if ever see "fail", "pass" otherwise)

- Can test in $O(1/\varepsilon \log n)$ time

(and can't do any better!)

An attempt:

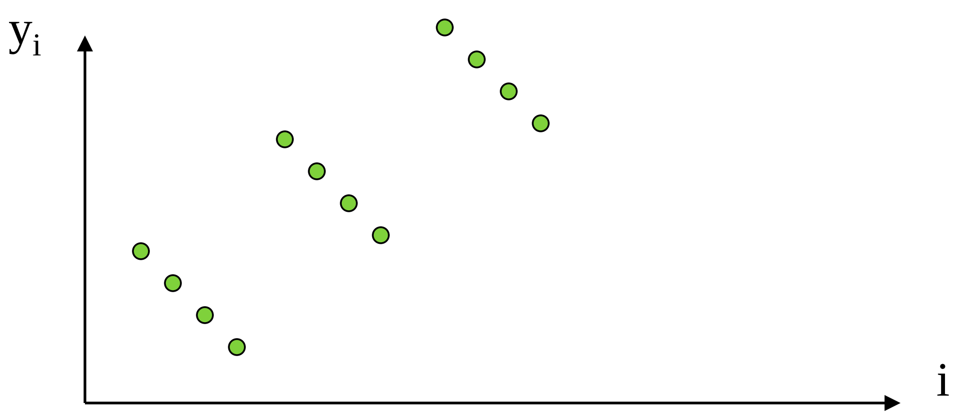
- Proposed algorithm:
  - Pick random i and test that $y_i \leq y_{i+1}$
- Bad input type:
  - 1,2,3,4,5,…n/4, 1,2,….n/4, 1,2,…n/4, 1,2,…,n/4
  - Difficult for this algorithm to find "breakpoint"
  - But other tests work well…



A second attempt:

- Proposed algorithm:
  - Pick random $i<j$ and test that $y_i \leq y_j$
- Bad input type:
  - *n/4* groups of 4 decreasing elements

    4,3, 2, 1,8,7,6,5,12,11,10,9…,4k, 4k-1,4k-2,4k-3,…
  - Largest monotone sequence is n/4
  - must pick *i,j* in same group to see problem
  - need $\Omega(n^{1/2})$ samples



A minor simplification:

- Assume list is distinct (i.e. $x_i \neq x_j$)
- Claim: this is not really easier
  - Why?

    Can "virtually" append *i* to each x_i

    $x_1, x_2, \ldots x_n \rightarrow (x_1, 1), (x_2, 2), \ldots, (x_n, n)$

    e.g., $1,1,2,6,6 \rightarrow (1,1),(1,2),(2,3),(6,4),(6,5)$

    Breaks ties without changing order

A test that works

[Ergun Kannan Kumar R Viswanathan]

- The test:

Test $O(1/\varepsilon)$ times:

  - Pick random i
  - Look at value of y_i
  - Do binary search for y_i
  - Does the binary search find any inconsistencies? If yes, FAIL
  - Do we end up at location i? If not FAIL

Pass if never failed

- Running time: $O(\varepsilon^{-1} \log n)$ time
- Why does this work?

Behavior of the test:

- Index *i* is good if binary search for y_i successful

- Test (restated):
  - pick $O(1/\varepsilon)$ *i*'s and pass if they are all good

$O(\frac{1}{\epsilon} \cdot \log n)$ time

- Correctness:
  - If list is sorted, then all i's good (uses distinctness) → test always passes
  - If list likely to pass test, then at least $(1-\varepsilon)n$ i's are good.
    - Main observation: good elements form increasing sequence
      - Proof: for i<j both good need to show $y_i < y_j$
        - let k = least common ancestor of i,j
        - Search for i went left of k and search for j went right of k → $y_i < y_k < y_j$
    - Thus list is ε-close to monotone (delete $< \varepsilon n$ bad elements)

Constructing a property tester:

- Find characterization of property that is
  - Efficiently (locally) testable
  - Robust -
    - objects that have the property satisfy characterization,
    - and objects far from having the property are unlikely to PASS

Usually the bigger challenge

More examples

- Can test if a function is a homomorphism in CONSTANT TIME (no dependence on domain size) [Blum Luby R.]

- Can test if the (sparse) social network has 6 degrees of separation in CONSTANT TIME [Parnas Ron]

Example: Homomorphism property of functions

- A "bad" testing characterization:

  $\forall x,y\ \ f(x)+f(y) = f(x+y)$

- Another bad characterization:

  For most x $\ f(x)+f(1) = f(x+1)$

- Good characterization ([Blum Luby R.]...):

  For most x,y $\ f(x)+f(y) = f(x+y)$

Warning:
Not true for all values of "most" –
Need at least 7/9 for general groups

Example: 6 degrees of separation

- Two “bad” testing characterizations:

  For every node, all other nodes within distance 6.

  For most nodes, all other nodes within distance 6.

- Good characterization [Parnas Ron]:

  For most nodes, there are many other nodes within distance 6.

Many more properties studied!

- Graphs, functions, point sets, strings, …
- Amazing characterizations of problems testable in graph and function testing models!

Properties of functions:

trigonometric, elliptic

low total degree polynomial

low comp

nta

lots more!

monotone

g

convex

submodular

affine invariant

Properties of graphs

- Dense graph properties:
  - completely characterized! (≈hereditary) [Alon Shapira] [Alon Fischer Newman Shapira] [Borgs Chayes Lovasz Sos Szegedy Vesztergombi]
- Hyperfinite graphs:
  - completely characterized! (all) … [Newman Sohler]
- General Sparse graphs:
  - bipartiteness, connectivity, diameter, colorability, expansion, rapid mixing, triangle free,… [Goldreich Ron] [Parnas Ron] [Czumaj Sohler] [Elek] [Batu Fortnow R. Smith White] [Kaufman Krivelevich Ron] [Alon Kaufman Krivelevich Ron]…

- Tools: Szemeredi regularity lemma, random walks, local search, simulate greedy, borrow from parallel algorithms

Some other combinatorial properties:

Sets:
Equality
Distinctness

Membership in low complexity languages:
regular
context-free
branching programs

Strings:
edit distance
compressibility

Codes:
BCH
Reed-Muller

etric properties;
clusterability
convex hull
embeddability

What else?

- Can we characterize the (constant time) testable properties?

“Classical” approximation

- Output number close to value of the optimal solution (not enough time to construct a solution)
- Some examples:
  - Minimum spanning tree,
  - vertex cover,
  - max cut,
  - positive linear program,
  - edit distance, …

A very simple example

Deterministic

Approximate answer

And (of course)…. sub-linear time!

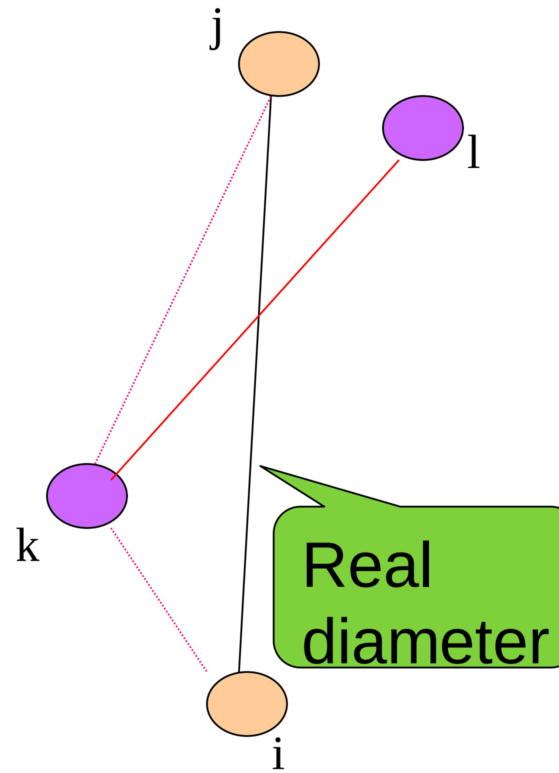
Approximate the diameter of a point set

- Given: m points, described by a distance matrix D, s.t.
  - D_{ij} is the distance from i to j.
  - D satisfies triangle inequality and symmetry.

  (note: input size $n = m^2$)
- Let i, j be indices that maximize D_{ij} then D_{ij} is the *diameter.*
- Output: k,l such that $D_{kl} \geq D_{ij}/2$

2-multiplicative approximation

Algorithm [Indyk]



- Algorithm:
  - Pick k arbitrarily
  - Pick l to maximize D_{kl}
  - Output D_{kl}
- Running time? $O(m) = O(n^{1/2})$
- Why does it work?

$D_{ij} \leq D_{ik} + D_{kj}$ (triangle inequality)

$\leq D_{kl} + D_{kl}$ (choice of l + symmetry of D)

$\leq 2D_{kl}$

Are there techniques that work for families of problems?

- Yes!
- We will see an example, but first, a slightly different model…

Large inputs
Large outputs

When we don’t need to see all the output…

do we need to see all the input?

Some examples

Locally decodable codes

Local property reconstruction

Local generation of random objects

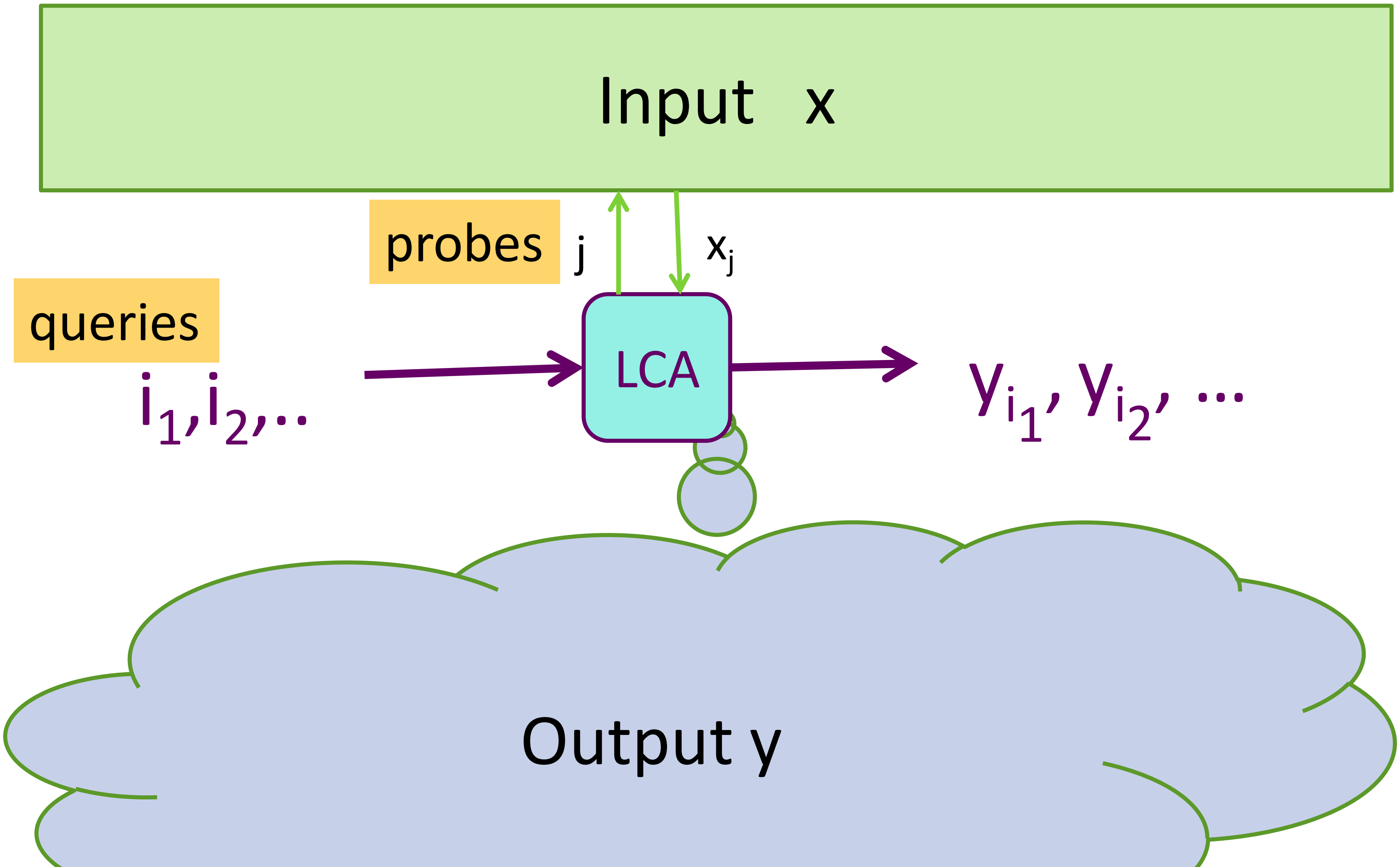
Estimating graph parameters: page rank, communities, dominating set, max matching …

Local decompression

A “unifying” model?

Local Computation Algorithms

[Alon R Tamir Vardi Xie]

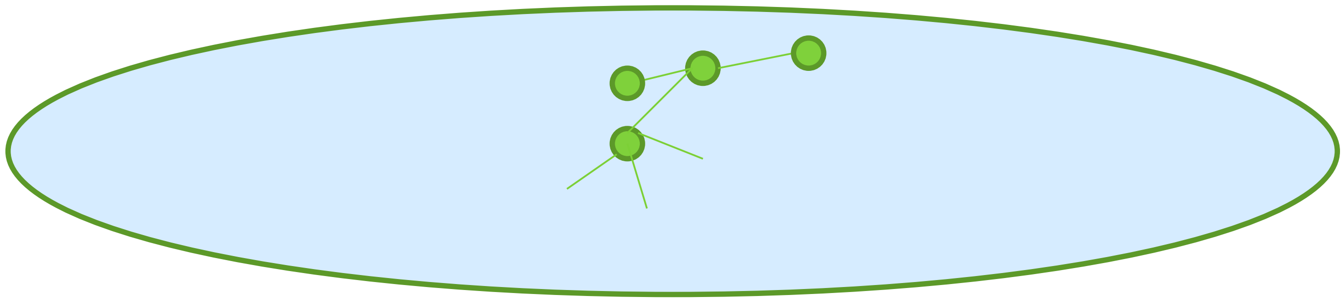


An example:

Maximal Independent Set

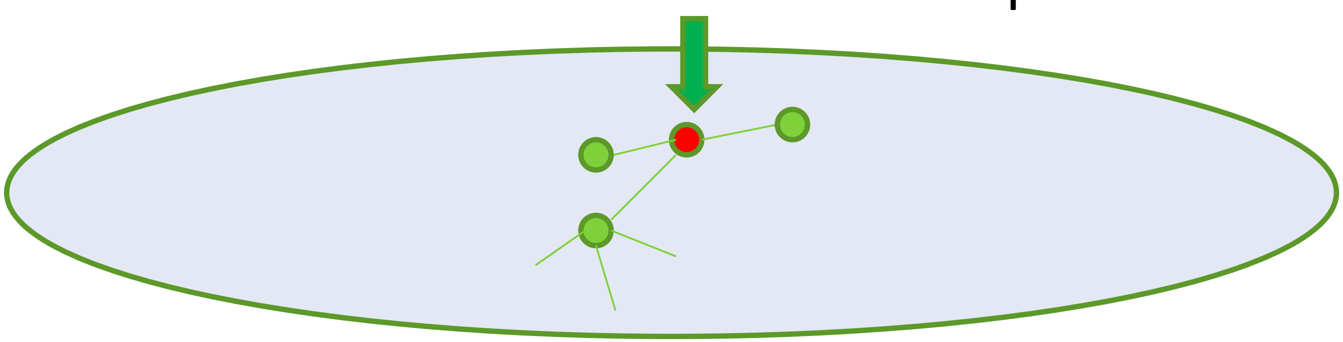
Maximal independent set

Input



Is node u in the maximal independent set?

Queries
To
Output



A fast local computation algorithm for bounded degree graphs

- Lazy Greedy Algorithm: (initially, MIS is empty)
  - Query: "Is node *u* in the MIS?"
  - Answer: if neighbors of *u* not in MIS, then put *u* into it (and remember decision!)

- Probe complexity: $O(d)$

Can we avoid these problems?

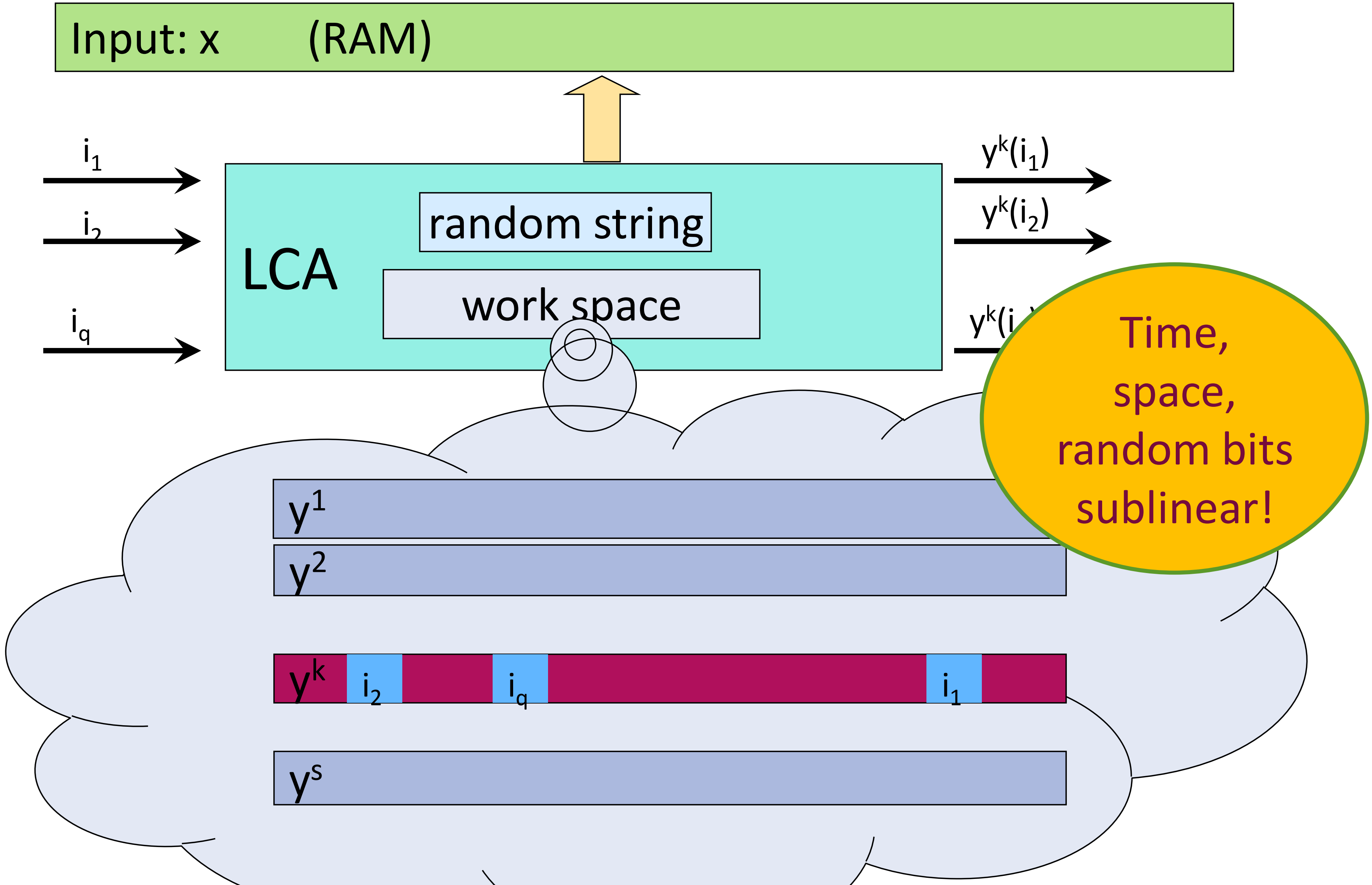
- Note:
  - $O(n)$ space to remember past choices
  - Answer depends on query order
  - Can't allow simultaneous non-interacting copies of LCA algorithm!!

A challenge:

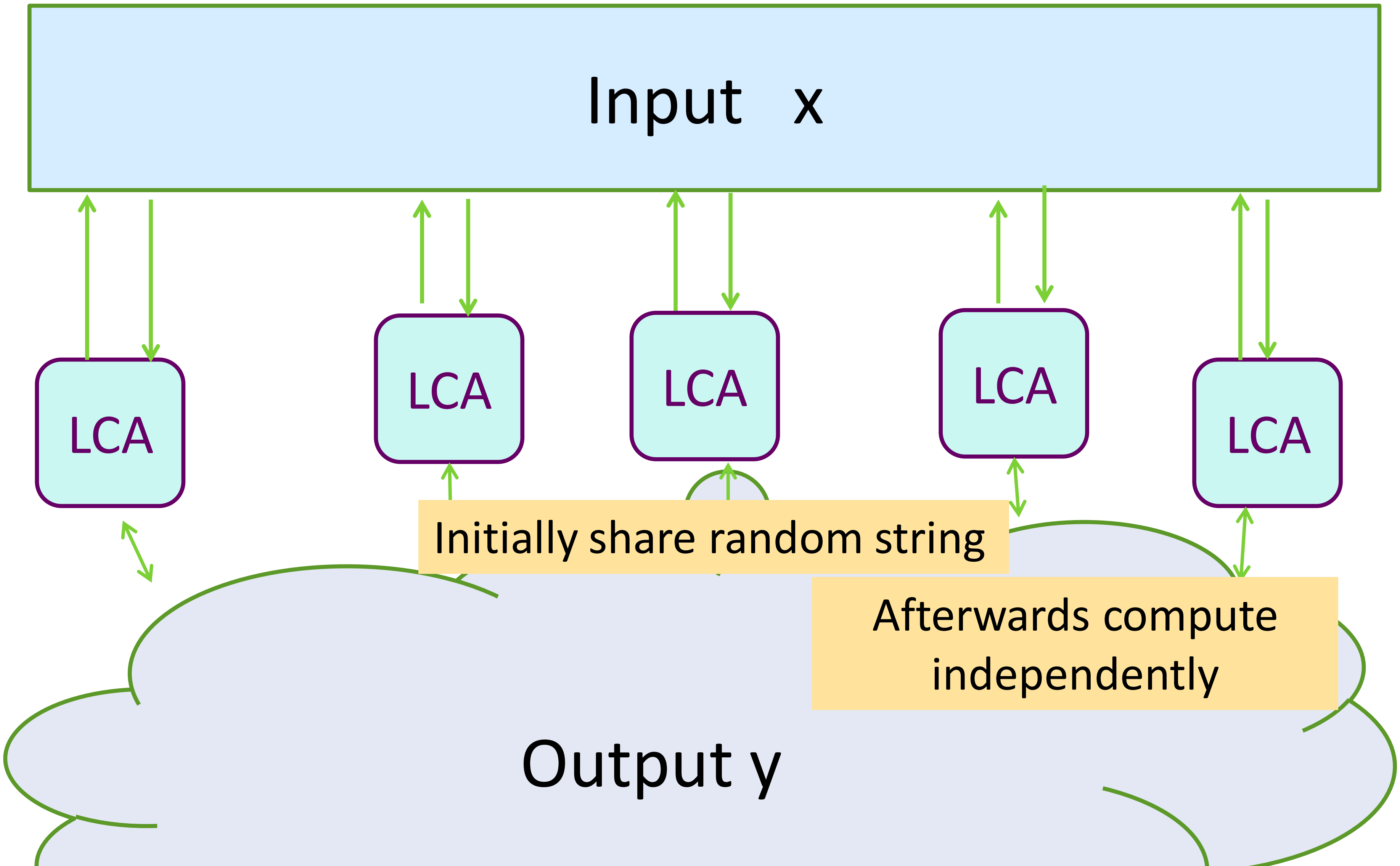
Consistency!

Local Computation Algorithms:
A model

Local computation algorithms



“Swarms” of LCAs



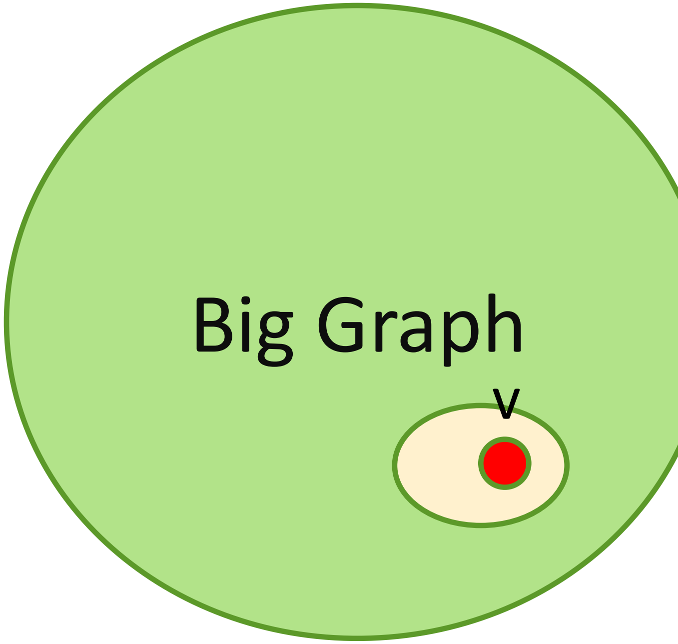
How do we design good LCAs?

A hope?

Find MIS algorithm *A* with nice property:

"any node v's output depends only on few inputs"

Then simulate A's behavior for v!



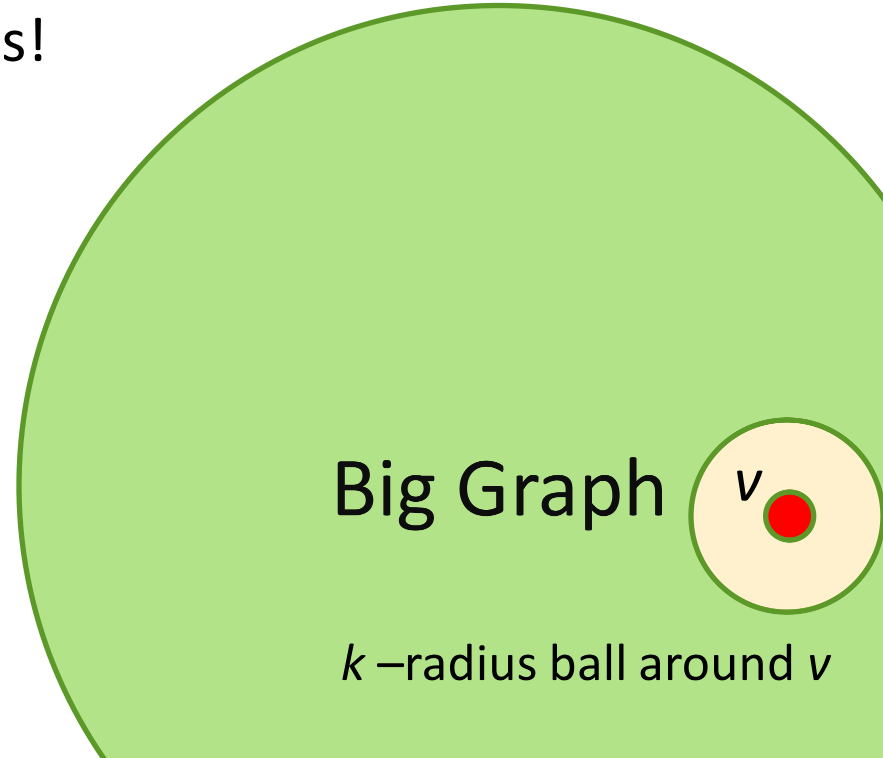
Idea 1:
Distributed Algorithms to the rescue!

Distributed algorithms give LCAs

[Parnas Ron]

- If there is a k round distributed algorithm for MIS, then:
  - v's output depends only on inputs and computations of k-radius ball around v
  - Can read/simulate in d^k probes!

- But how big is k?



Local *distributed* algorithms

In this context:

Local = Constant rounds

fantastic progress in local distributed algorithms!!!

How fast can MIS be computed in a distributed setting?

- Lexicographically-first-MIS is P-complete [Cook]

- Randomized O(log n) rounds [Luby]
  - Yields $d^{c \log n} = 2^{O(\log d \log n)}$ time LCA

- With additional ideas/different algorithms, can do a lot better and solve several other problems!!

[Barenboim Elkin] [R Tamir Vardi Xie] [Alon R Vardi Xie] [Barenboim Elkin Pettie Schneider] [Even Medina Ron][Reingold Vardi][Chung Pettie Su] [Levi R Yodpinyanee] [Ghaffari]…

Ideas from LCAs also used in improved distributed algorithms!

Idea 2:
LCAs via Simulating GREEDY

Simulating GREEDY

[Nguyen Onak]… [Alon R. Vardi Xie]

- Simulate sequential GREEDY
  - Run through nodes in some order
  - Put v in MIS if none of neighbors in MIS yet
- LCA computes: “What would GREEDY do on *u*?”
  - Must simulate results of greedy for all adjacent edges/nodes of lower ordering
  - Dependency chains can be long?
    - Most nodes ok if order is RANDOM! [NO]
    - We need more than “most”

Random order greedy

- Dependency chains are short [ARVX]
  - Galton-Watson branching processes
- Short random seed is enough
  - log n-wise independence

How fast can LCAs for MIS be?

- Dependence on n?
  - [R. Tamir Vardi Xie][Alon R. Vardi Xie] [Reingold Vardi] [Levi R. Yodpinyanee] $\mathrm{poly}\log n$
  - [Even Medina Ron] $\log^* n$
- Dependence on d?
  - [R. Tamir Vardi Xie] [Alon R. Vardi Xie] [Even Medina Ron] [Reingold Vardi] EXPONENTIAL
  - [Levi R. Yodpinyanee] $2^{c\log^3 d} \log^3 n$
  - [Ghaffari] $2^{c\log^2 d} \log^3 n$

OPEN QUESTION:
Can we get *poly(d)* dependence?

Some other LCA results:

- Approximate maximum matching, bipartite weighted vertex cover [Mansour Vardi] [Even Medina Ron] [Feige Mansour Schapire]

  Polynomial in d [Levi R. Yodpinyanee]

  Used in learning setting [Feige Mansour Schapire]
- Radio network broadcast scheduling [RTVX]
- Graph, Hypergraph coloring [RTVX] [Feige Patt-Shamir Vardi] [Czumaj Mansour Vardi]
- k-CNF [RTVX]
- Local computation mechanism design [Hassidim Mansour Vardi]
- Online algorithms [Mansour Rub
  - load balancing balls a

Polylog query and space complexity

Back to sublinear approximations ….

Example: Approximate maximum matching

- If you have an LCA for approximate maximum matching M
- Algorithm to estimate size of M:
  - Sample several edges uniformly
  - ask LCA which edges in M?
  - Output (fraction of edges in M)x(total number of edges)

General paradigm:

LCA → sublinear time approximation

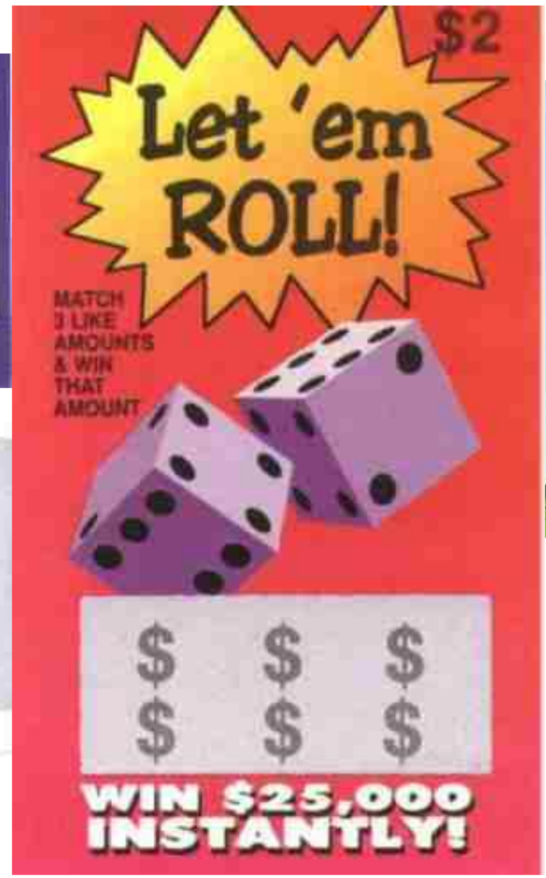
[Parnas Ron]

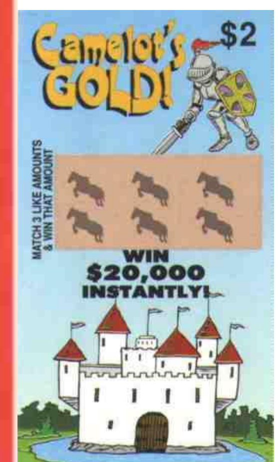
Part II

No samples

What if data only accessible via random samples?

Play the lottery?





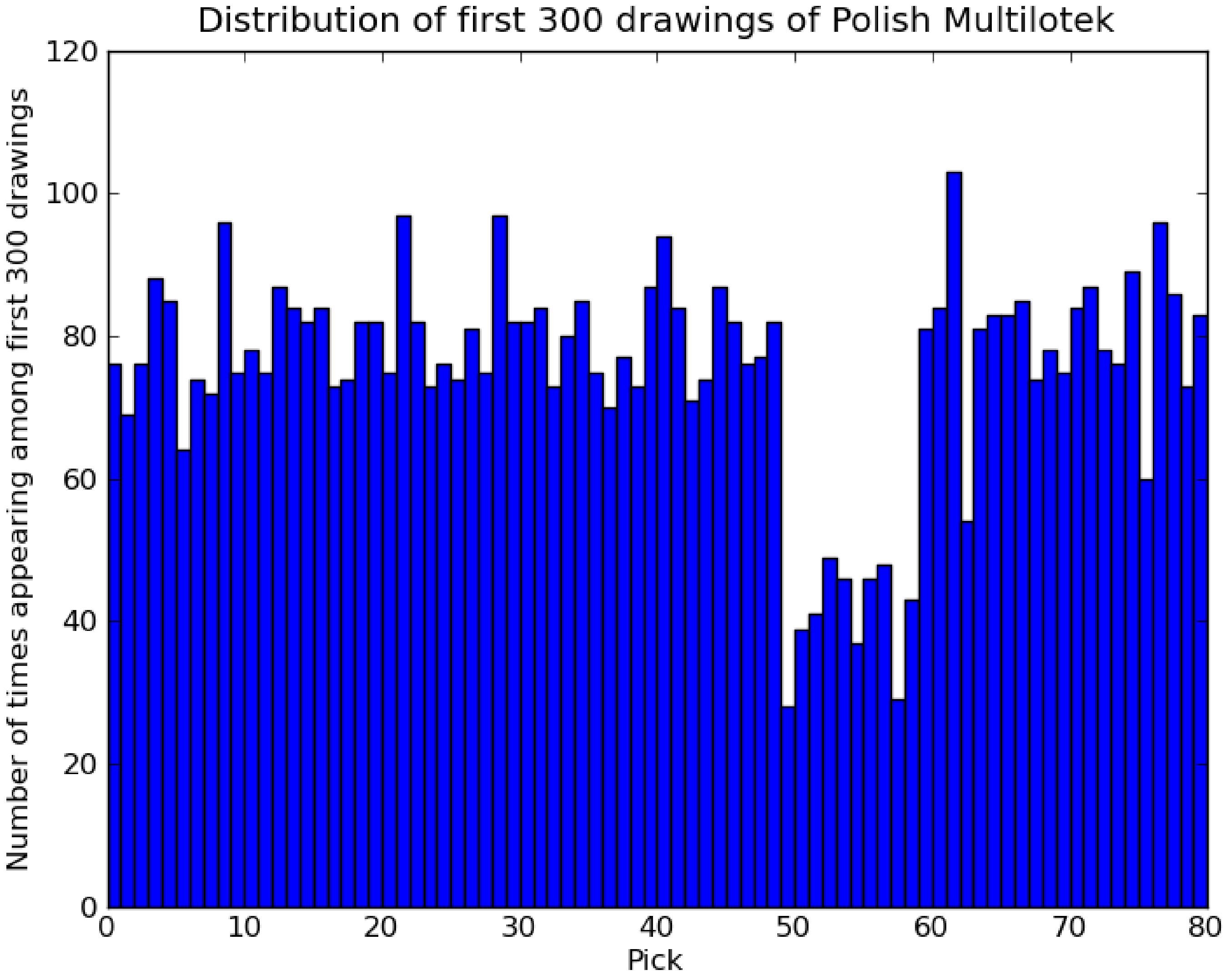
Is the lottery unfair?

- From Hitlotto.com: Lottery experts agree, past number histories can be the key to predicting future winners.

True Story!

- Polish lottery Multilotek
  - Choose “uniformly” at random distinct 20 numbers out of 1 to 80.
  - Initial machine biased
    - e.g., probability of 50-59 too small
- Past results:

http://serwis.lotto.pl:8080/archiwum/wyniki_wszystkie.php?id_gra=2



Thanks to Krzysztof Onak (pointer) and Eric Price (graph)

Distributions on BIG domains

- Given samples of a distribution, need to know, e.g.,
  - entropy
  - number of distinct elements
  - “shape” (monotone, bimodal,…)
  - closeness to uniform, Gaussian, Zipfian…
  - Ability to generate the distribution?
- Do we need assumptions on shape of distribution?
  - i.e., smoothness, monotonicity, normal distribution,…
- Considered in statistics, information theory, machine learning, databases, algorithms, physics, biology,…

Key Question

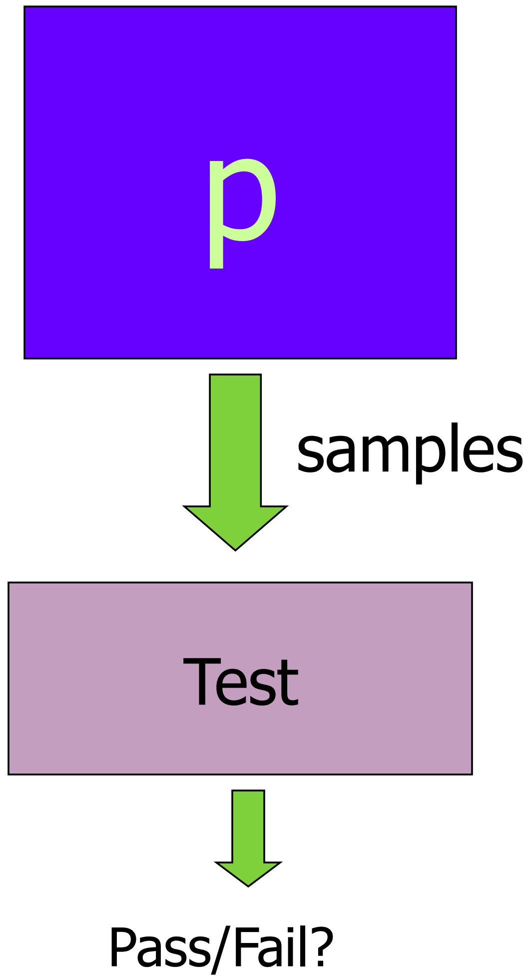
- How many samples do you need in terms of *domain size*?
  - Do you need to estimate the probabilities of each domain item?

  -- OR --

  - Can sample complexity be *sublinear* in size of the domain?

The model

Our usual model:



- *p* is *arbitrary* black-box distribution over *[n]*, generates iid samples.
- p_i = Prob[*p* outputs *i*]
- Sample complexity in terms of *n*?

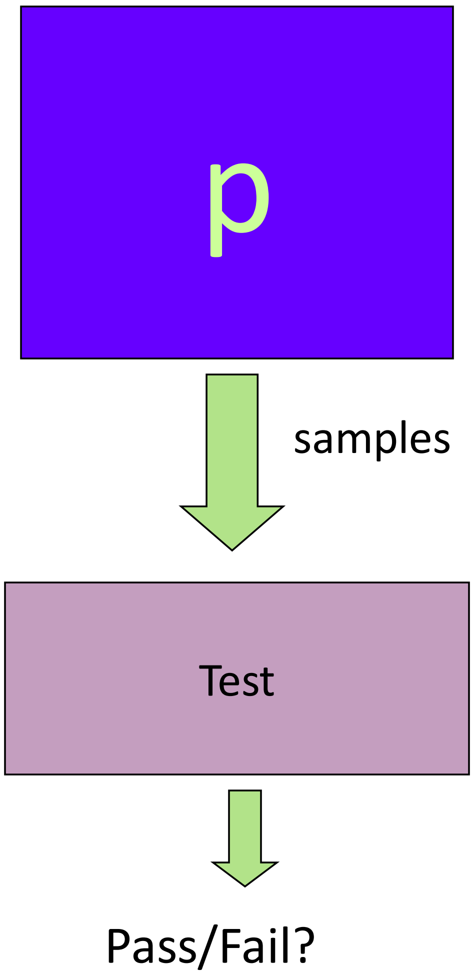
A first set of properties:

Similarity of distributions

Similarities of distributions

- Are p and q close or far?
  - q is known to the tester ("goodness of fit")
    - q is uniform
  - q is given via samples

Is p uniform?



Sample complexity of distinguishing

$$p = U$$

from $||p - U||_1 > \varepsilon$

is $\theta(n^{1/2})$

An idea: [Goldreich Ron]

- L_2 distance (squared): $||p-q||_2^2 = \sum(p_i - q_i)^2$
- $||p\text{-}U||_2^2 = \Sigma(p_i - 1/n)^2$
  $= \Sigma p_i^2 - 2\Sigma p_i/n + \Sigma 1/n^2$
  $= \Sigma p_i^2 - 1/n$

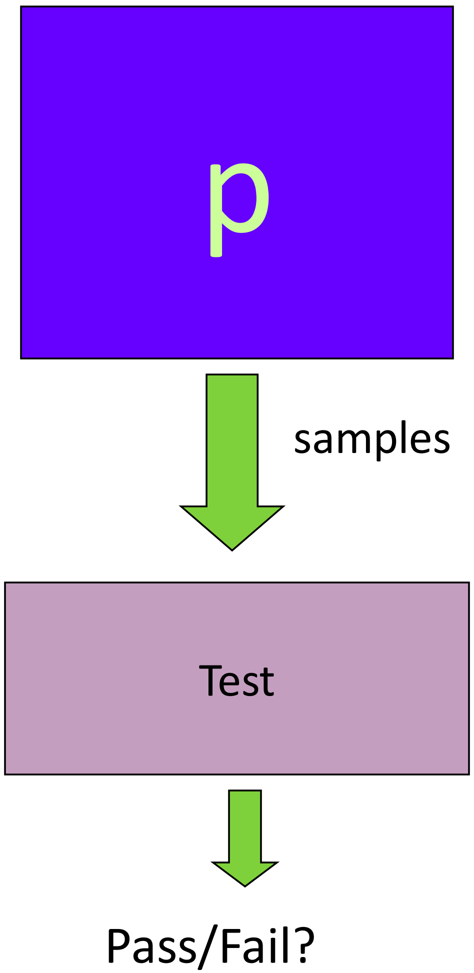
Minimized for uniform distribution

- Estimate collision probability to estimate L_2 distance from uniform

Uniformity Testing History

- [Goldreich Goldwasser Ron] $\Omega(n^{\alpha})$ lower bound
- [Goldreich-Ron] (implicit): $O(\frac{\sqrt{n}}{\epsilon^4})$ upper bound via collision probability
- [Batu Fortnow Rubinfeld Smith White]: $\Omega(\sqrt{n})$ lower bound (+ explicit upper bound)
- [Paninski '03]: upper bound of $O(\frac{\sqrt{n}}{\epsilon^2})$, assuming $\epsilon = \Omega\left(n^{-\frac{1}{4}}\right)$ via number distinct elements. Lower bound of $\Omega(\frac{\sqrt{n}}{\epsilon^2})$.
- [Chan Diakonikolas Valiant Valiant] [Diakonikolas Kane Nikishkin] Similar to χ^2- based. Optimal for all settings.
- [Diakonikolas Gouleakis Peebles Price '16] Collision based tester also optimal!
- [Diakonikolas Gouleakis Peebles Price '17] nontrivial p-values!

Is *p* uniform?



- Sample complexity of distinguishing

$$p = U$$

from $||p - U||_1 > \varepsilon$ is $\theta(n^{\frac{1}{2}})$

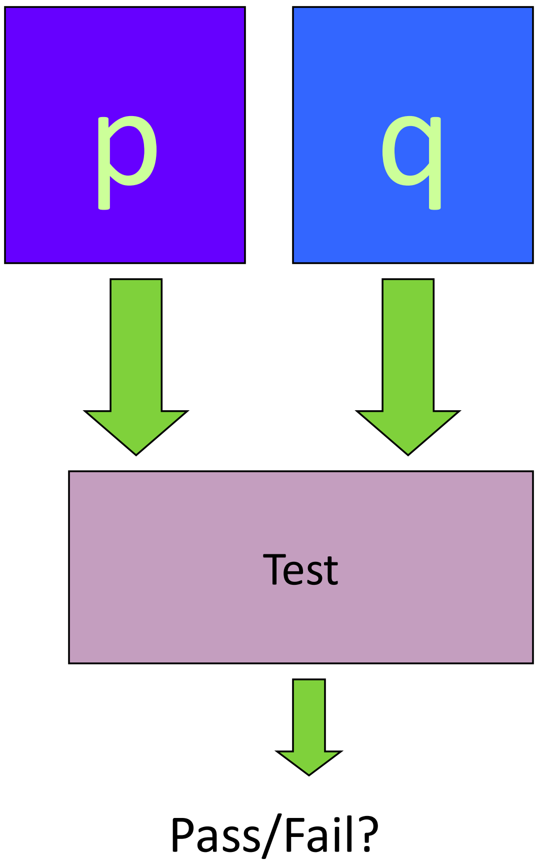
- Same complexity to test if p is any *known* distribution

"Testing identity"

Identity Testing History:

- [Batu Fischer Fortnow Kumar R. White] $O\left(\sqrt{n}\, polylog(n)\epsilon^{-4}\right)$ (collisions) Reduce to uniformity testing via grouping similar probability elements in q.
- [Onak]: running time matches sample complexity
- [Valiant Valiant, Diakonikolas Kane Nikishkin]: $O\left(\sqrt{n}/\epsilon^{2}\right)$ (uses chi-squared like tester)
- [Diakonikolas Kane] Simpler bucket-avoiding reduction. Simpler and general lower bound paradigm.
- [Goldreich] Reduction to uniformity testing with same complexity.

Testing closeness



Theorem: Sample complexity of distinguishing

$$p = q$$

from $||p - q||_1 > \epsilon$

is $\theta(n^{\frac{2}{3}})$

Why so different?

- Collision statistics are all that matter
- Collisions on “heavy” elements can hide collision statistics of rest of the domain
- Construct pairs of distributions where heavy elements are identical, but “light” elements are either identical or very different

Closeness between unknown distributions

- [Batu Fortnow R. Smith White]: $O(\frac{n^{\frac{2}{3}}\log(n)}{\epsilon^{\frac{8}{3}}})$ upper bound for testing closeness between two unknown discrete distributions. Candidate lower bound family.

- [P. Valiant]: lower bound of $\Omega\left(n^{\frac{2}{3}}\right)$ for constant error.

- [Chan Diakonikolas Valiant Valiant]: tight upper and lower bound of $O(\max\{\frac{n^{\frac{2}{3}}}{\epsilon^{\frac{4}{3}}}, \frac{n^{\frac{1}{2}}}{\epsilon^{2}})$

- [Diakonikolas Kane] simpler lower bound, upper bound. Upper bound beats worst case in large class of instances.

Approximating the distance between two distributions?

Distinguishing whether

$||p - q||_1 < \varepsilon$ or $||p \ - q||_1 \ > \varepsilon'$

requires $\theta(\frac{n}{\log n})$ samples

[P. Valiant 08, G. Valiant P. Valiant 11,

Wu Yang 14, Han Jiao Weissman 15]

Independence

Independence of pairs

- p is *joint distribution* on pairs *<a,b>* from *[n] x [m]*
  (wlog *n≥m*)

- For marginal distributions $p_1, p_2,$
  p *independent* iff $p = p_1 \times p_2$

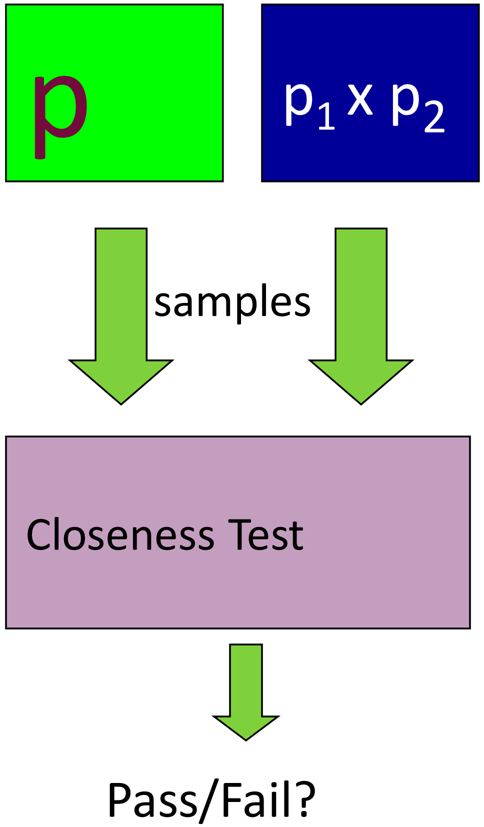
- "Robustness" Lemma [Sahai Vadhan]
  If $||p - p_1 \times p_2||_1 > \epsilon$ then $\forall\, A, B \; ||p - A \times B||_1 > \epsilon/3$

1st try: “Naïve” Algorithm

- Algorithm:
  - Approximate marginal distributions $f_1 \approx p_1$ and $f_2 \approx p_2$
  - Use Identity testing algorithm to test that $p \approx f_1 x f_2$
- Number of queries: O$(n+m + (nm)^{1/2})$
  - But, if support of p_1 is bounded from below by b, then can do O$(1/b + m + (nm)^{1/2})$
  - *(also note: if n=m, then this is very good!)*

A difficulty – “tolerant testing” setting

2^{nd} idea: use closeness test



- Simulate p_1 and p_2, and check $||p - p_1 \times p_2||_1 > \epsilon$
- Behavior:
  - If $p = p_1 \times p_2$ then PASS
  - If $||p - p_1 \times p_2||_1 > \epsilon$ then FAIL
  - Sample complexity: $O((nm)^{2/3})$
    - Better if max probability element is bounded from above!

Independence testing

[Batu Fischer Fortnow Kumar R. White]:

$O(n^{\frac{2}{3}}m^{\frac{1}{3}} \cdot polylog\, n \cdot poly\left(\frac{1}{\epsilon}\right))$ upper bound. Candidate lower bound family.

[Levi Ron R.]:

lower bounds for constant error $\Omega(m^{\frac{1}{2}}n^{\frac{1}{2}})$ and $\Omega(n^{\frac{2}{3}}m^{\frac{1}{3}})$ for $n = \Omega(m \log m)$

[Acharya Daskalakis Kamath]: upper bound of $O\left(\frac{n}{\epsilon^2}\right)$for $n = m$.

[Diakonikolas Kane] matching bound of $\theta(\max\left\{n^{\frac{2}{3}}m^{\frac{1}{3}}\epsilon^{-\frac{4}{3}}, \frac{(mn)^{\frac{1}{2}}}{\epsilon^2}\right\})$, optimal bounds for all dimensions

Information theoretic quantities

Entropy

Support size

Compressibility

Can we get *multiplicative* approximations for entropy?

- In general, no….
  - ≈0 entropy distributions are hard to distinguish

2-approx. in $n^{\frac{1}{4}}$

- What if entropy is bigger?
  - Can γ-multiplicatively approximate the entropy with $\tilde{O}(n^{1/\gamma^2})$ samples (when entropy $>2\gamma/\varepsilon$) [Batu Dasgupta R. Kumar]
  - requires $\Omega(n^{1/\gamma^2})$ [Valiant]
  - better bounds when support size is small [Brautbar Samorodnitsky]
  - Similar bounds for estimating support size [Raskhodikova Ron R. Smith] [Raskhodnikova Ron Shpilka Smith]

Additive approximations for entropy and support size

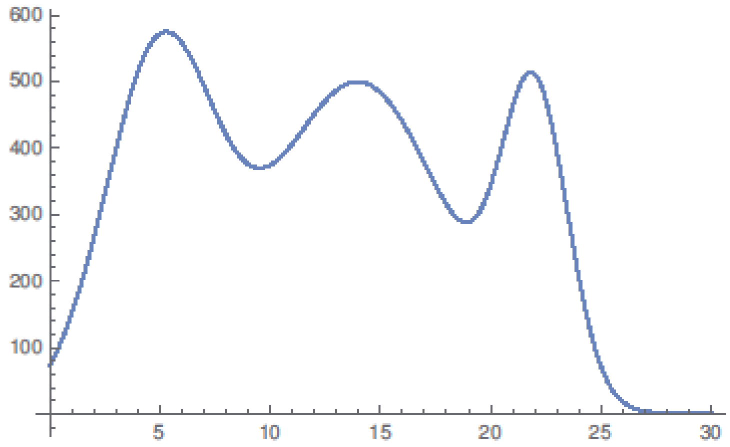
need $\theta(n/\log n)$ samples [Raskhodnikova Ron Shpilka Smith] [Valiant] [Valiant Valiant][Wu Yang] [Han Jiao Weissman]

Properties of high dimensional spaces:

- Limited independence: [Alon Andoni Kaufman Matulef R Xie] [Haviv Langberg]
- Monotonicity over general posets [Batu Kumar R] [Bhattacharyya Fischer R P. Valiant] [Acharya Daskalakis Kamath]
- Junta *distributions* [Aliakbarpour Blais R]
- Bayesian Networks [Canonne Diakonikolas Kane Stewart] [Daskalakis Pan]
- Ising Models [Daskalakis Dikkama Kamath]
- Joint properties of many distributions – similar distributions, clustering distributions, similar means [Levi Ron R. 2011, Levi Ron R. 2012, Diakonikolas Kane, Aliakbarpour Blais R. 2016]

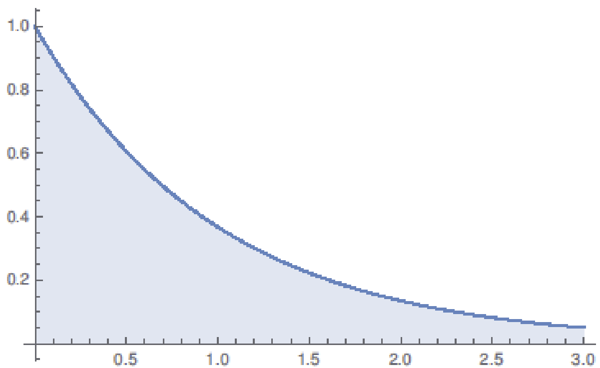
AND MORE AND MORE!!!

Testing via shape

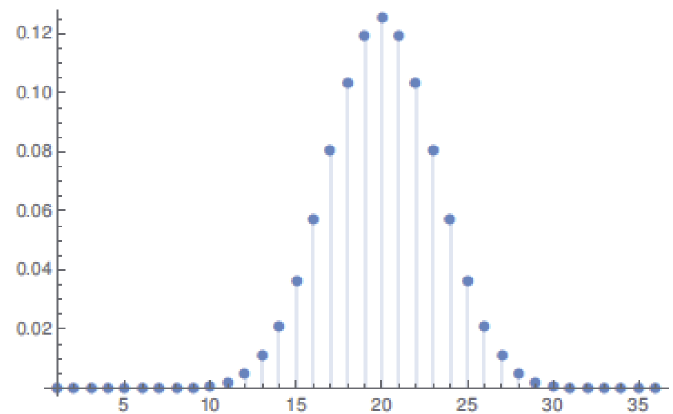


Some distribution families defined by shape:

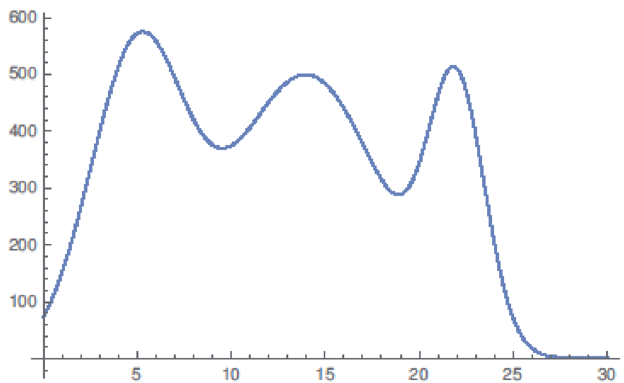
Monotone



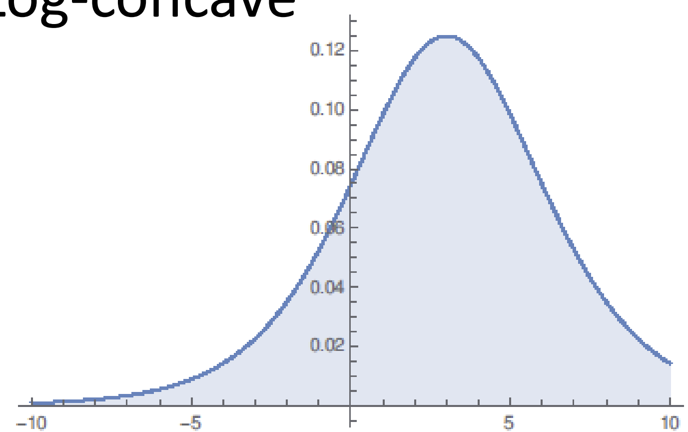
Poisson Binomial (PBD)



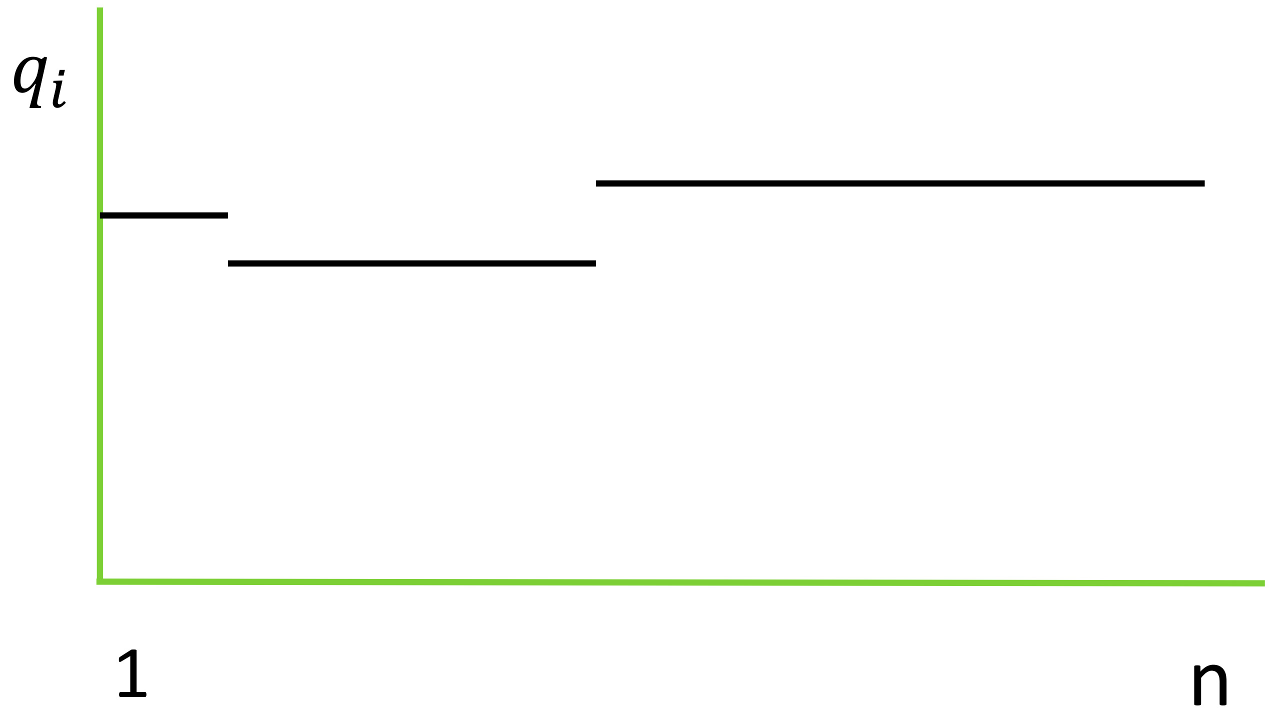
t-modal



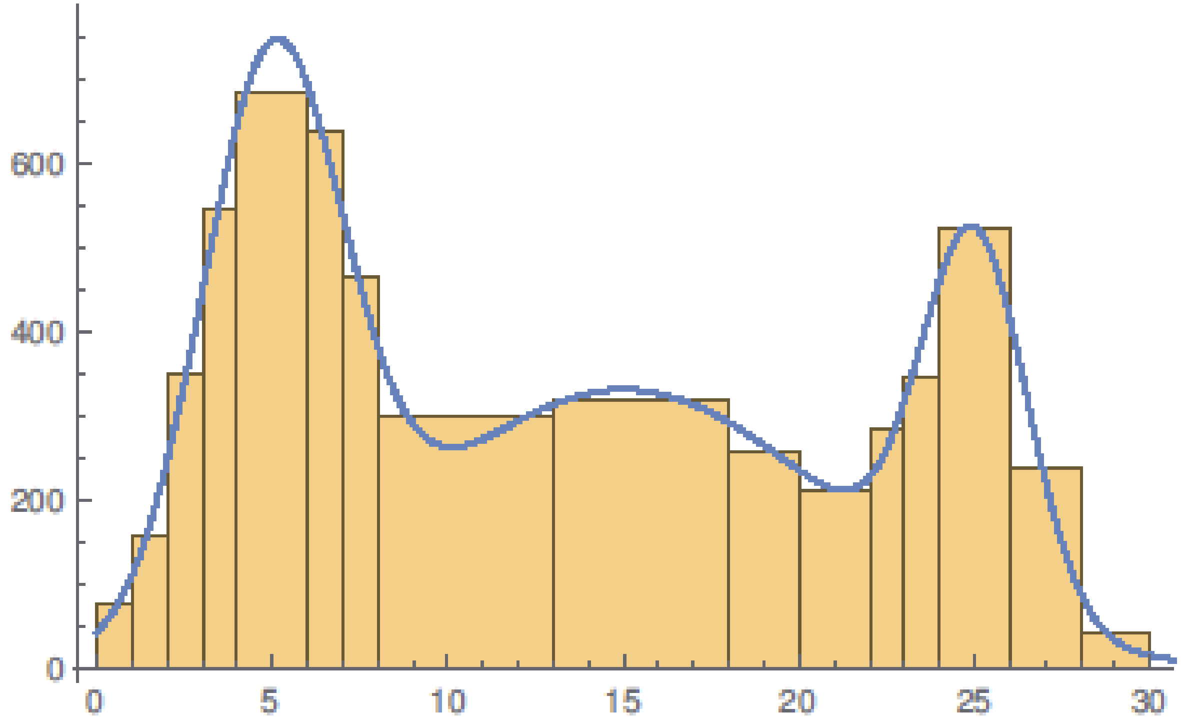
Log-concave



Another one:
k-flat (*k*-histogram, k-piecewise constant) distributions



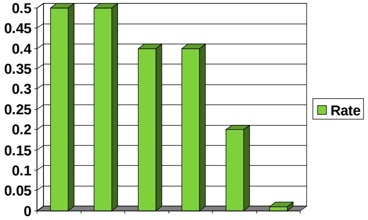
Use k-histograms to approximate?



Example: Monotone (nonincreasing) distributions

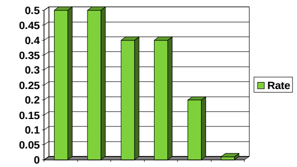
Monotone distributions over *totally ordered domains [1..n]:*

$i<j$ implies $p_i \geq p_j$



Lower bound [Batu Kumar R.]

Lemma: Testing monotonicity requires $\Omega(\sqrt{n})$ samples



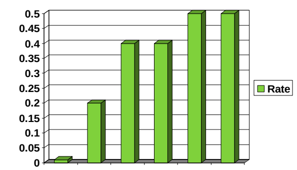
p

Proof:

p close to uniform

iff

p, p^R = "reversal" of p, are both close to monotone



p^R

Upper bounds for monotonicity testing?

$O(\sqrt{n}\ log(n))$ samples

[Batu Kumar R][Daskalakis Diakonikolas Servedio]

Birge Buckets for Monotone Distributions [Birge][Daskalakis Diakonikolas Servedio]

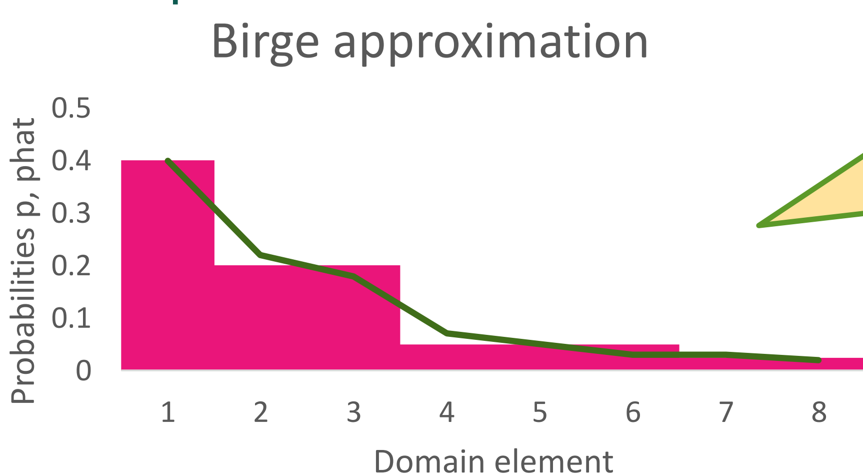
Partition of domain into buckets (segments) of size $(1+\epsilon)^i$

($O(\frac{1}{\epsilon}\log n)$ buckets total)

oblivious

For distribution p, let $\hat{p}$ be such that uniform on each bucket, but same conditional probability in each bucket

Then $||p-\hat{p}|| \leq \epsilon$



Enough to learn the weights of each bucket

Test Monotonicity

- Approximate distribution by *(log n)-flat* distribution:
  - Questions:
    - Is each bucket close to uniform?
    - Total weights of each bucket?
- Check if *(log n)*-flat distribution close to monotone
  - Solve linear program

Generic algorithm idea:

- Approximate distribution by *k-flat* distribution:
  - Questions:
    - Does it exist for small k?
    - How do you find interval boundaries?
- Check if k-flat distribution close to class
  - Solve linear program?

Can't assume the distribution has the property

General testing paradigm

[Canonne Diakonikolas Gouleakis R.]

Monotone, k-modal, log-concave, Monotone Hazard Rate, Binomial, Poisson Binomial, k-histograms, k-piecewise degree d polys, k-Sums of independent integer random values

- [Batu Kumar R.] [Daskalakis Diakonikolas Servedio] [Daskalakis Diakonikolas Servedio Valiant Valiant] monotonicity, k-modal
- [Chan Diakonikolas Servedio Sun] piecewise poly learning
- [Levi Indyk R.] [Acharya Diakonikolas Hegde Li Schmidt] k-histogram
- [Acharya Daskalakis] Discrete Gaussians
- [Daskalakis Diakonikolas O'Donnell Servedio Tan][Diakonikolas Kane Stewart] optimal SIIRV learning
- [Canonne] more testing improvements

See also

[Acharya Daskalakis Kamath] (different paradigm, same classes)

Many other properties to consider!

- Higher dimensional flat distributions
- Mixtures of *k* Gaussians
- Generated by a small Markovian process
- …

Dependence on *n*

- *o(n)*
- But usually n^{α} for some $0 < \alpha < 1$

Is this good or bad?

nontrivial

but still daunting!

Getting past the lower bounds

- Restricted classes of distributions
  - Structured distributions [Batu Dasgupta Kumar R] [Batu Kumar R] [Servedio R] [Daskalakis Diakonikolas Servedio Valiant Valiant] [Diakonikolas Kane Nikishkin]
  - Competitive closeness testing [Acharya Das Jafarpour Orlitsky Pan Suresh] [Valiant Valiant 14] [Diakonikolas Kane 16]
- Other distance measures
- More powerful query models (see survey [Canonne])

Conclusion:

- Distribution testing problems are everywhere
- For many problems, we need a lot fewer samples than one might think!
- Many COOL ideas and techniques have been developed
- Lots more to do!

Thank you