Streaming Algorithms for Halo Finders

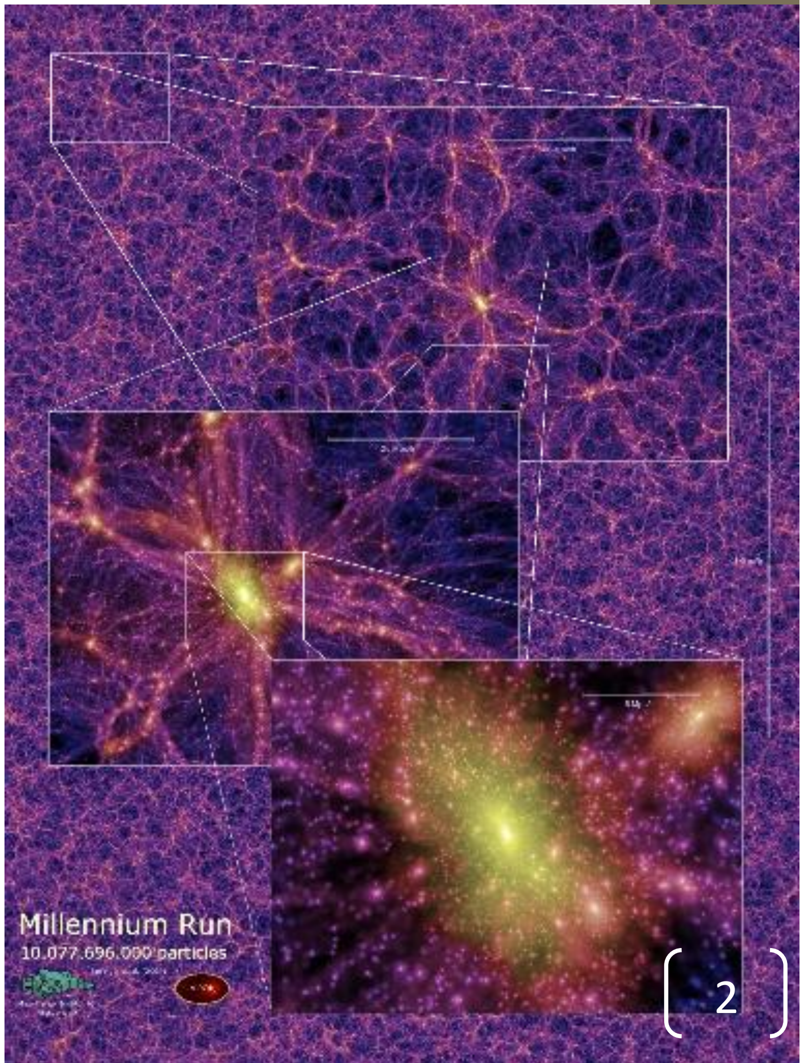
Zaoxing Liu , **Nikita Ivkin** , Lin F. Yang , Mark Neyrinck ,
Gerard Lemson, Alexander S. Szalay, Vladimir Braverman,
Tamas Budavari, Randal Burns, Xin Wang

Johns Hopkins University, Baltimore, MD, USA

Cosmological Simulations

Simulation:

- is a gravitational evolution of the system of particles
- provides distribution of particles in space and time
- helps to understand the processes of forming galaxies



Cosmological Simulations

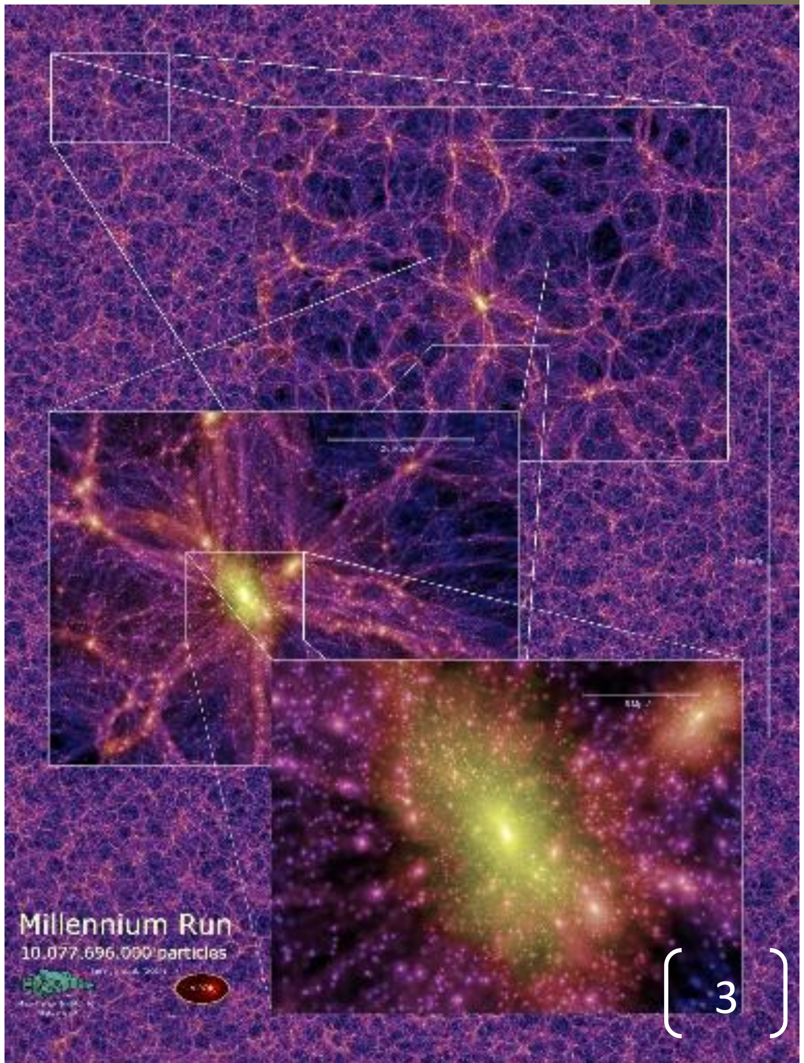
What can we observe in reality?

- macro-structures, such as galaxies and patterns of galaxies

What can we measure and compare?

- macro structures from simulation and reality
- macro structures from different simulations

Extraction of macro structures is crucial to connect theory to observation.



Halo

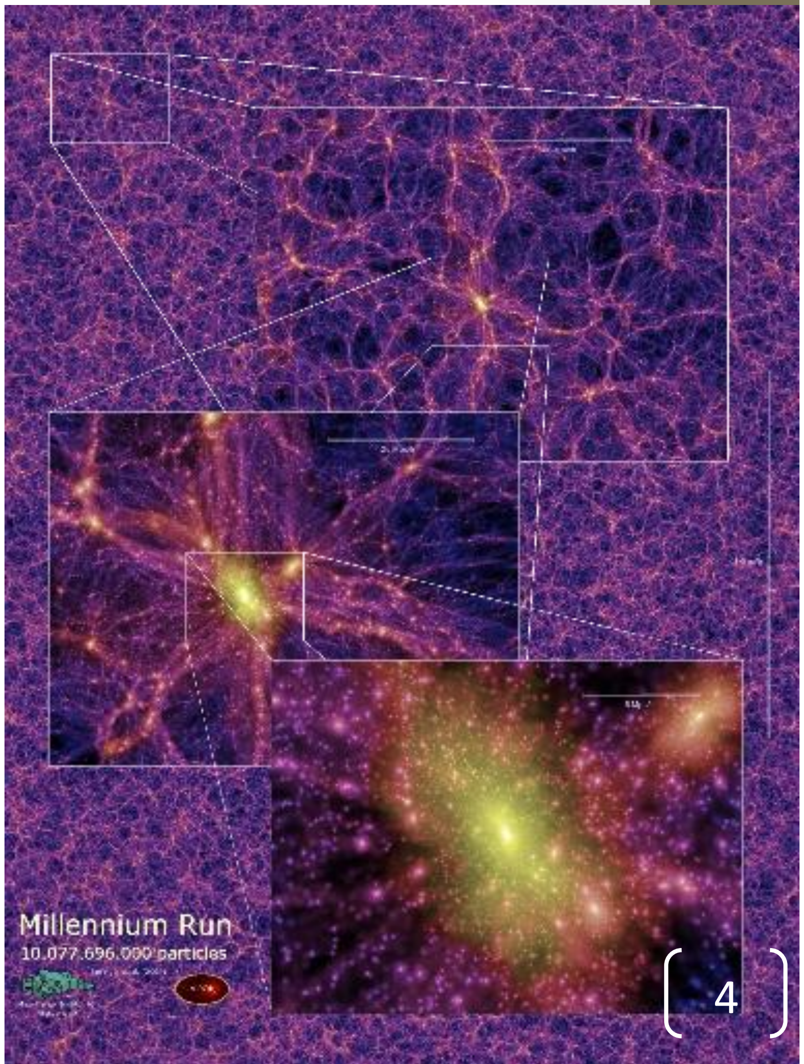
In terms of Physics:

- Galaxies are thought to form in halos

Defining property:

- Macro structure with high mass concentration

There is no agreed-upon formal definition of a halo.

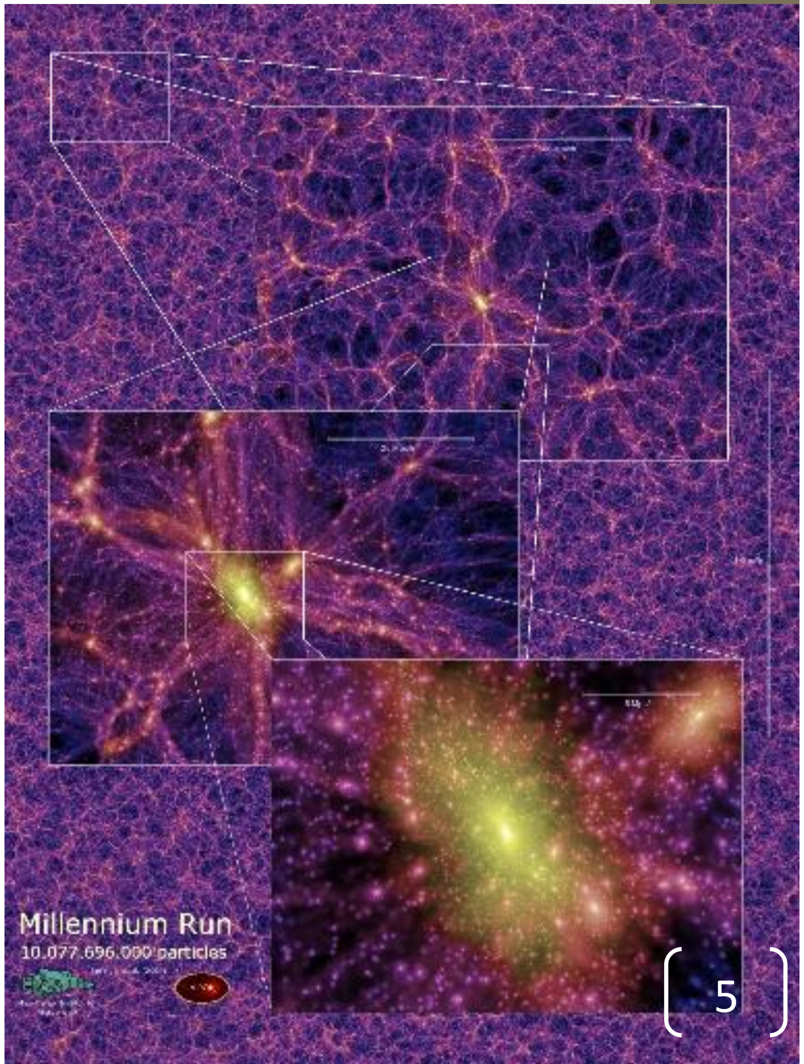


Halo

There is no agreed-upon formal definition of a halo.

We can not introduce absolute measure how good is certain halo finder

We can introduce measure to compare outputs of different halo finders



Halo

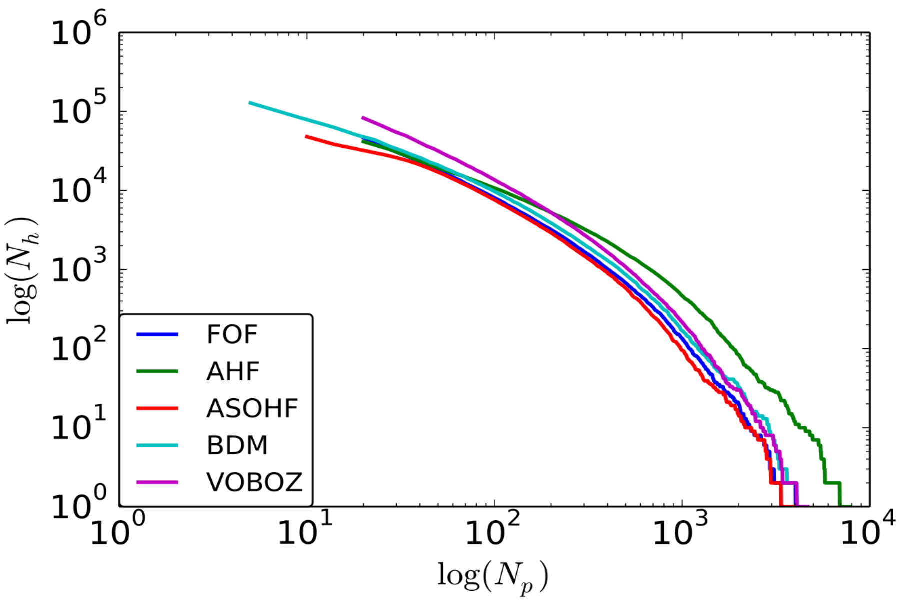
Some facts about data:

- Total number of particles: $\sim 10^{12}$
- Number of Haloes: $\sim 10^{9}$
- Particles not associated with halos: ~ 80-90%
- Particles not associated with large halos: ~ 99.9%

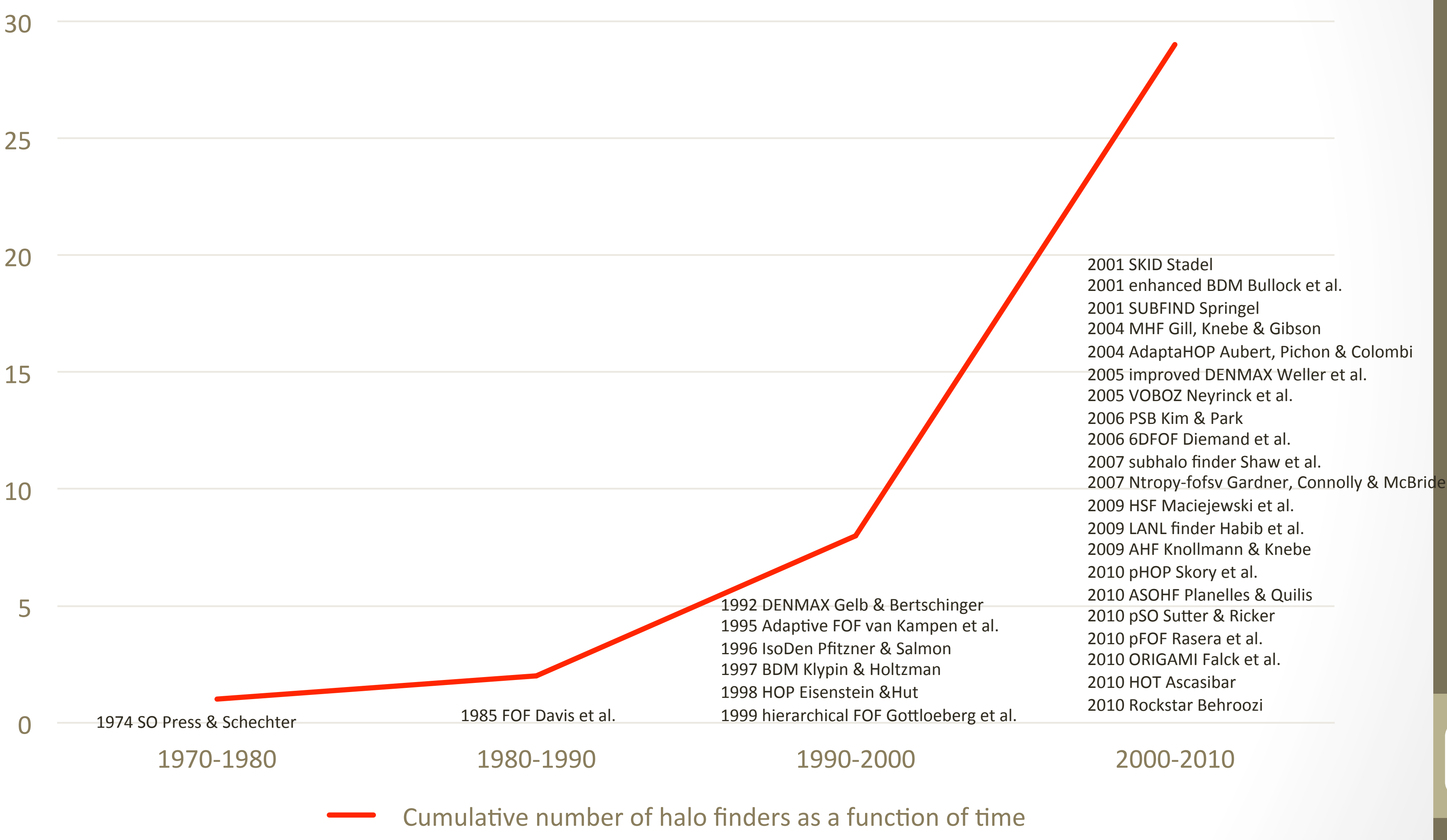
Distribution of halos sizes

N_h - number of haloes,

N_p - number of particles in halo



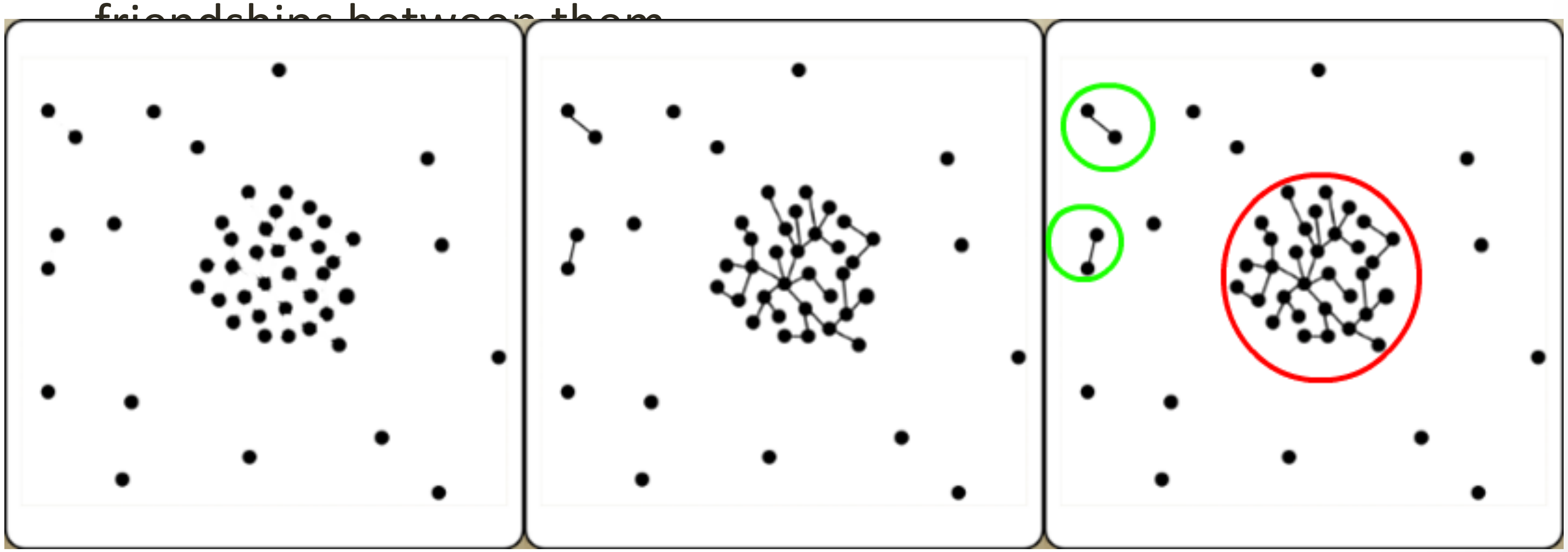
Halo finding algorithms



The Halo-Finder Comparison Project
[Knebe et al, 2011]

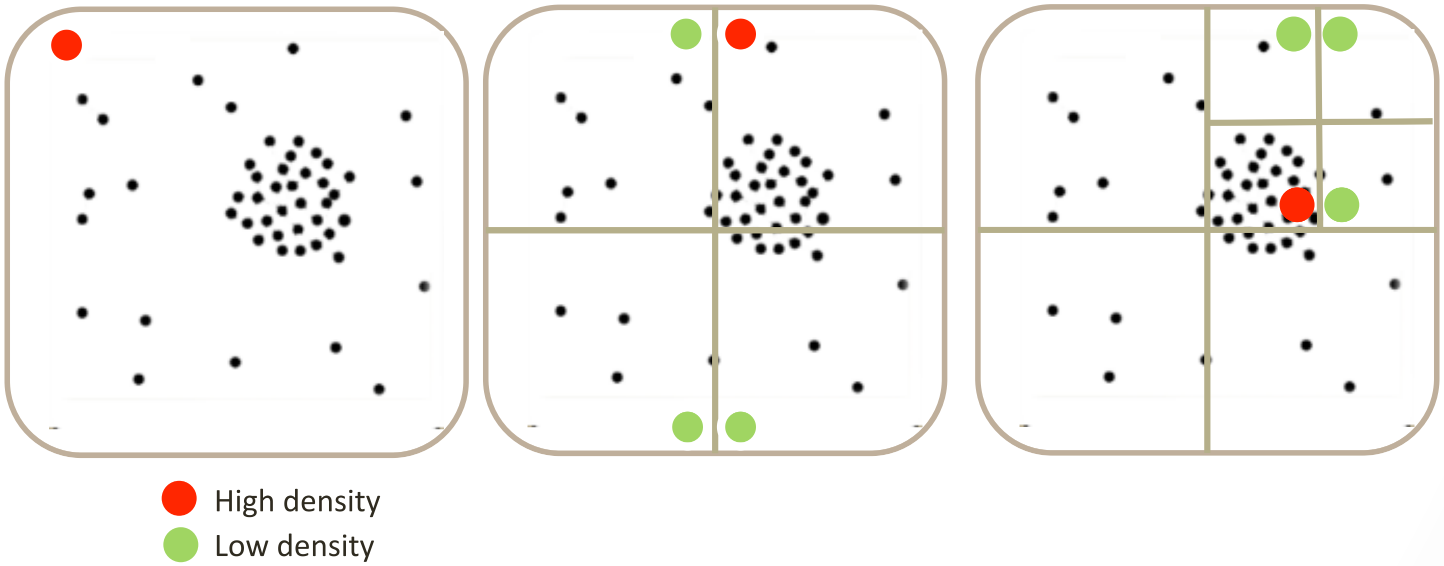
Friends-of-Friends Algorithm

- FOF is one of the very first halo finding algorithms [Davis et al, 1985]
- Simple conceptually, is the first step in many other algorithms
- Has a single free parameter called the linking length θ.
  - Two particles are "friends" if the distance between them less than θ.
  - Two particles are in the same cluster if there exists a chain of friendships between them



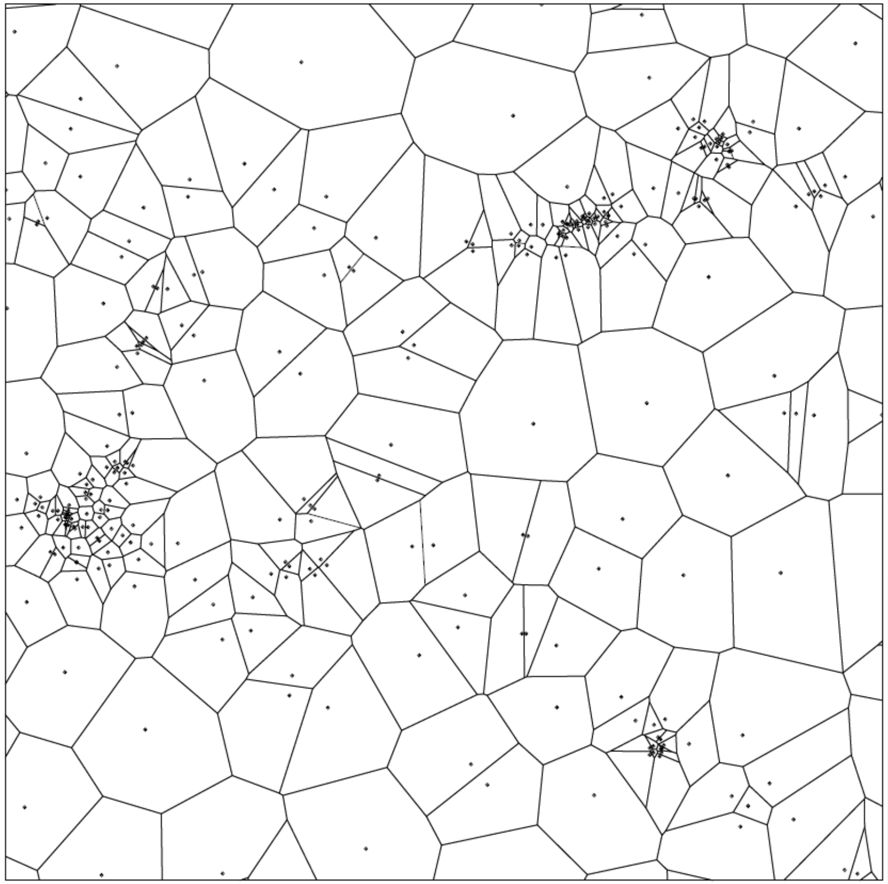
AHF

- **A**MIGA's **H**alo **F**inder [Knollmann&Knebe, 2009]:
    1. Estimate densities in the regular grid
    2. Find overdense cells according to given threshold
    3. Build subgrid in each cell with high density, and iterate



VOBOZ

- **VO**ronoi **BO**und **Z**ones [Neyrinck et al, 2004]:
  1. Create Voronoi Tessellation
  2. Measure the density at each particle, based on size of Voronoi cell
  3. Group particles around density maxima;



Memory issue

All current halo finders requires to load all the data into memory

Each time snapshot from the simulation with 10^{12} particles will require 12 terabytes of memory

To build a scalable solution we need to develop an algorithm with sublinear memory usage

Streaming algorithms:

Motivational Example

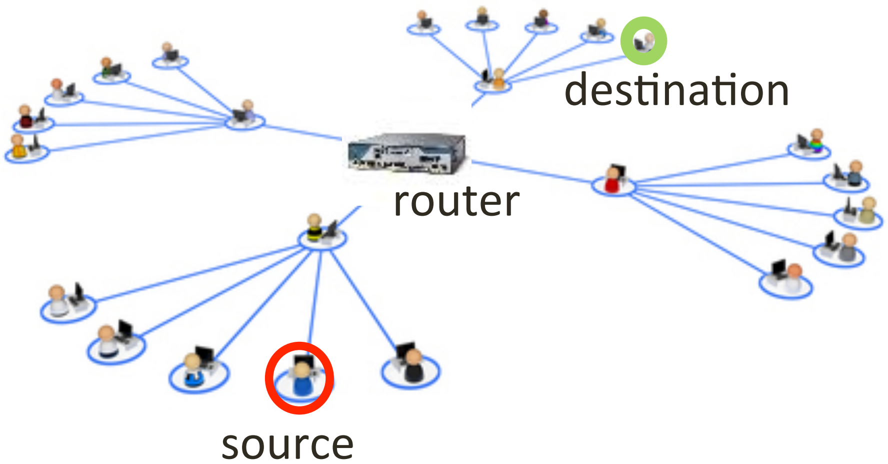
Network traffic analysis

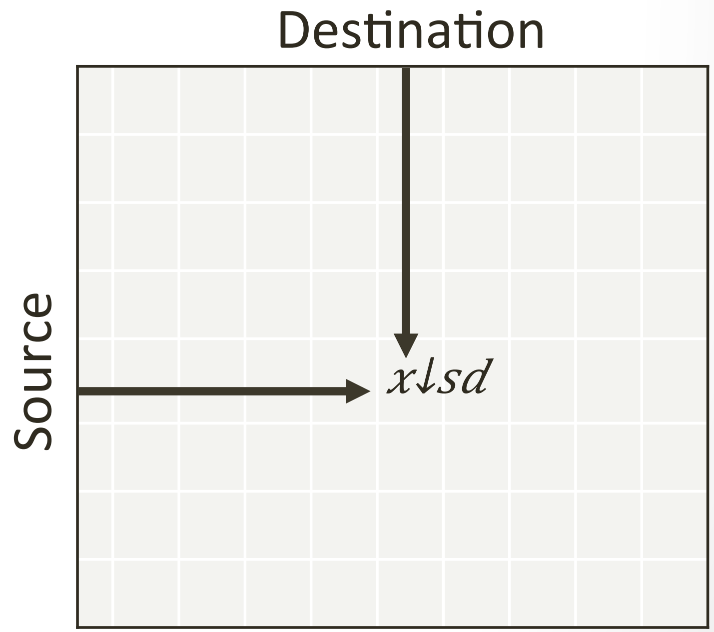
Goal: maintain Source/Destination statistics on data packets going through node (router)

Naive solution: store matrix of counters

for each (sd) packet increment x_{sd}

Issue: space ($2^{32} \times 2^{32}$ entries)





Streaming model

Stream: m elements from dictionary of size n

e.g. $D=\{x_1\ x_2 \ldots x_m\} = 3\ 5\ 3\ 7\ 5\ 4 \ldots$

Goal: Compute a function of stream e.g. median, number of distinct elements, longest increasing sequence, top k most frequent elements, etc.

Restrictions:

1. Limited working memory: sublinear in n and m
2. Access data sequentially, small number of passes
3. Process each element quickly

But approximate answers with high probability is OK.

Streaming problems

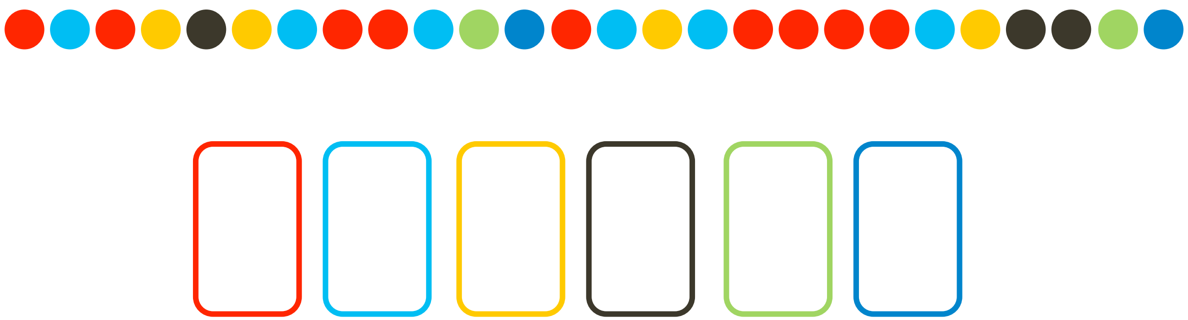
Frequency moment estimation:

$$f_i = |\{ j : x_j = i \;\; x_j \in D \}|$$

For each element i we define frequency f_i as the number of its occurrences in the stream D.

$$F_k = \sum_{i=1}^{n} f_i^k$$

Then F_k is ***k*-th frequency moment** of the stream.



f_i: 9 6 4 3 2 2

$F_2 = 9^2 + 6^2 + 4^2 + 3^2 + 2^2 + 2^2 = 150$

Streaming problems

Heavy hitters search:

We will say that i-th element of stream is (F_2, α)-heavy if

$$\boxed{f_i^2 \geq \alpha F_2}$$

Then ε-approximate (α, F_2)-heavy hitter problem is to find a set of elements T:

$$\forall i \in \{1,\dots,n\},\ f_i^2 > \alpha F_2 \Rightarrow i \in T.$$

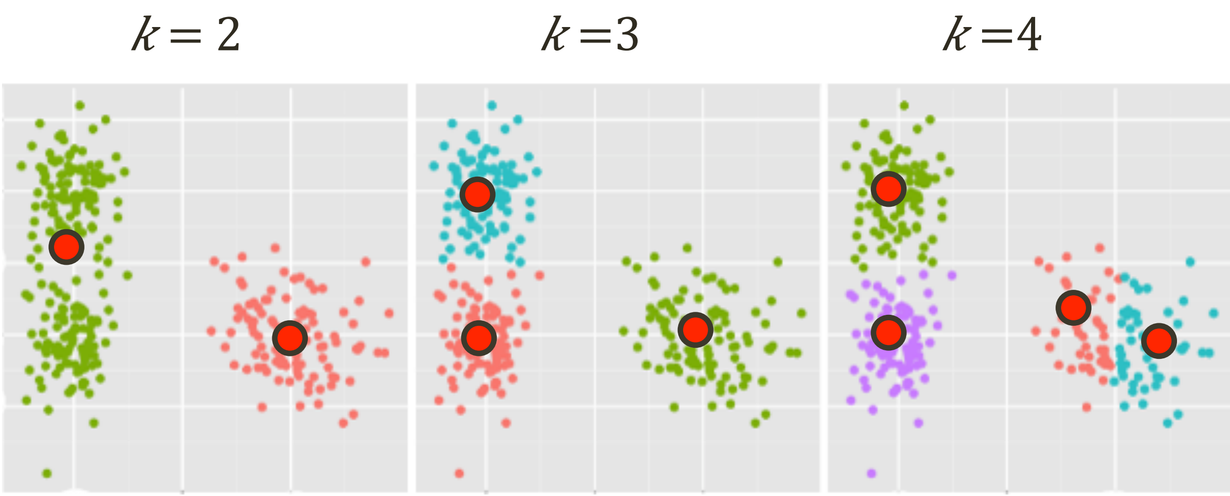
$$\forall i \in \{1,\dots,n\},\ f_i^2 < (\alpha - \varepsilon) F_2 \Rightarrow i \notin T.$$

Streaming problems

k-median:

Given a stream of points find a set of k centers $\{c_i\}_{i=1}^{k}$, which minimize cost function:

$$Q(C) = \sum_{j=1}^{m} \min_{c_i} (dist(c_i, x_j))$$



Streaming Solution:

Our goal:

- Reduce halos finding problem to one of the existing problems in streaming setting
- Apply ready-to-use algorithms

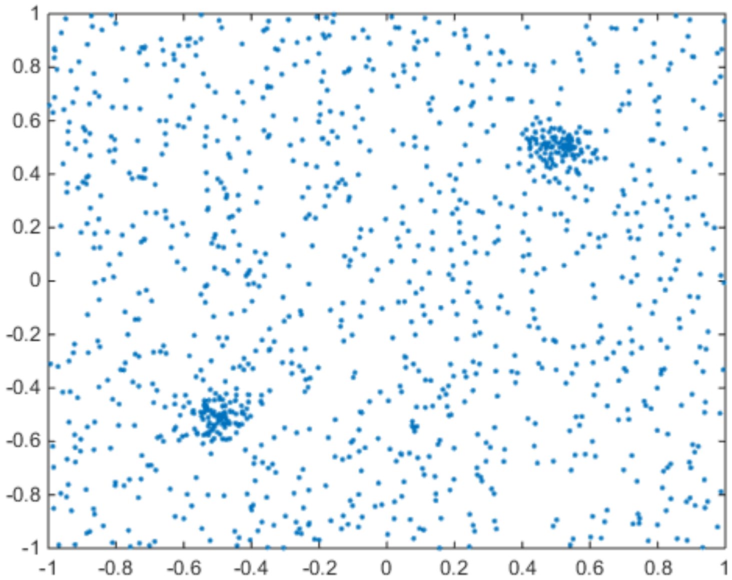
haloes $\approx$ k-median clusters?

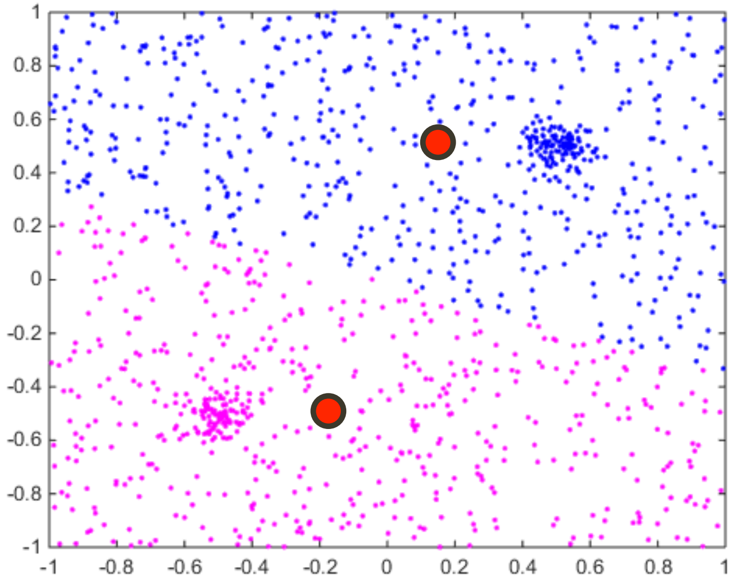
- There is no ready-to-use k-median clustering algorithm for problem where number of particles that are not assigned to any of clusters is so high ($\sim$ 90%)

Streaming Solution:

haloes ≈ k-median clusters?

- There is no ready-to-use k-median clustering algorithm for problem where number of particles that are not assigned to any of clusters is so high (~ 90%)





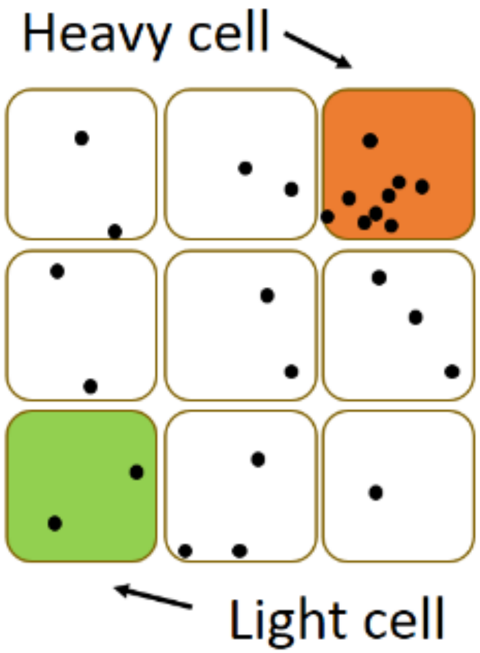
Streaming Solution:

Our goal:

- Reduce halos finding problem to one of the existing problems in streaming setting
- Apply ready-to-use algorithms

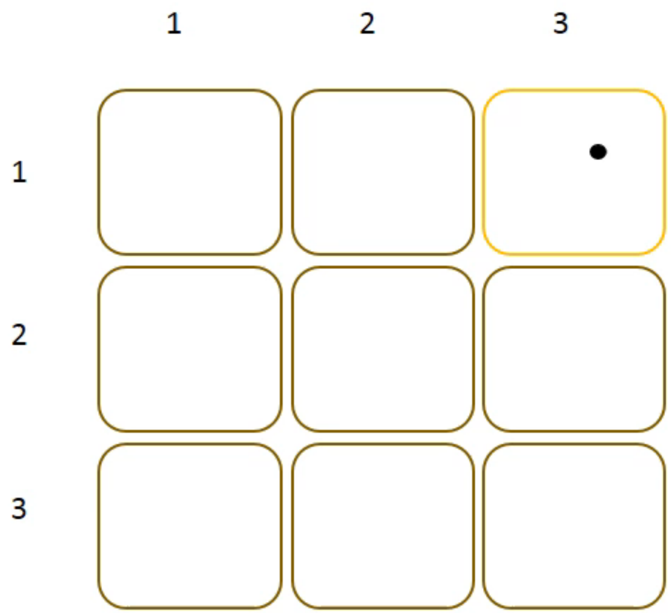
haloes ≈ heavy hitters?

- To make a reduction to heavy hitters we need to discretize the space.
- Naïve solution is to use 3D mesh:
  - Each particle now replaced by cell id
  - Heavy cells represent mass concentration
  - Grid size is chosen according to typical halo size



Streaming Solution:

haloes ≈ heavy hitters?



(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)	(3,1)	(3,2)	(3,3)
0	0	1	0	0	0	0	0	0

Heavy Hitter Streaming Algorithms

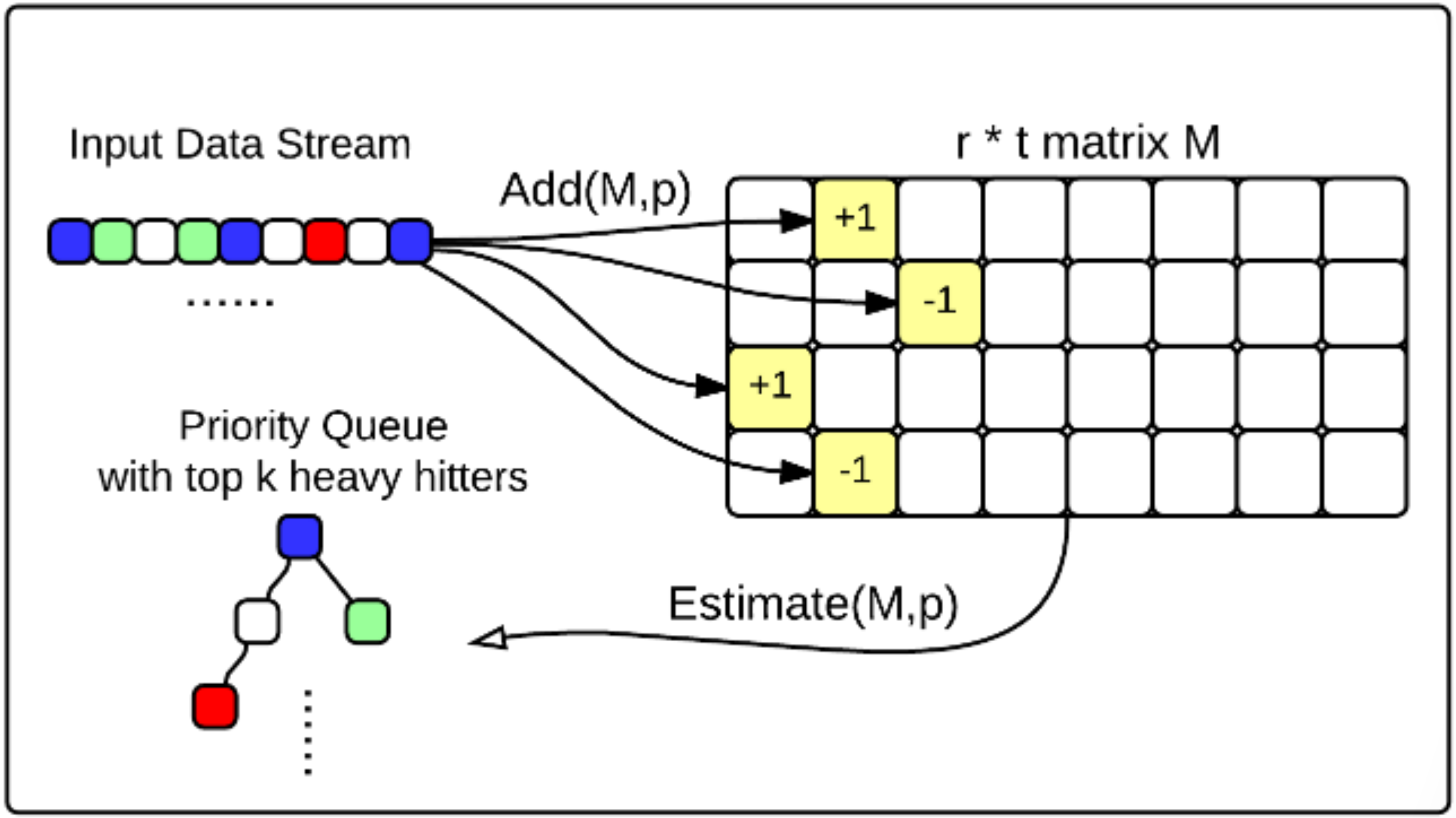
Count-Sketch Algorithm:

- Maintain a sketch of the data stream to approximate the heavy hitters.

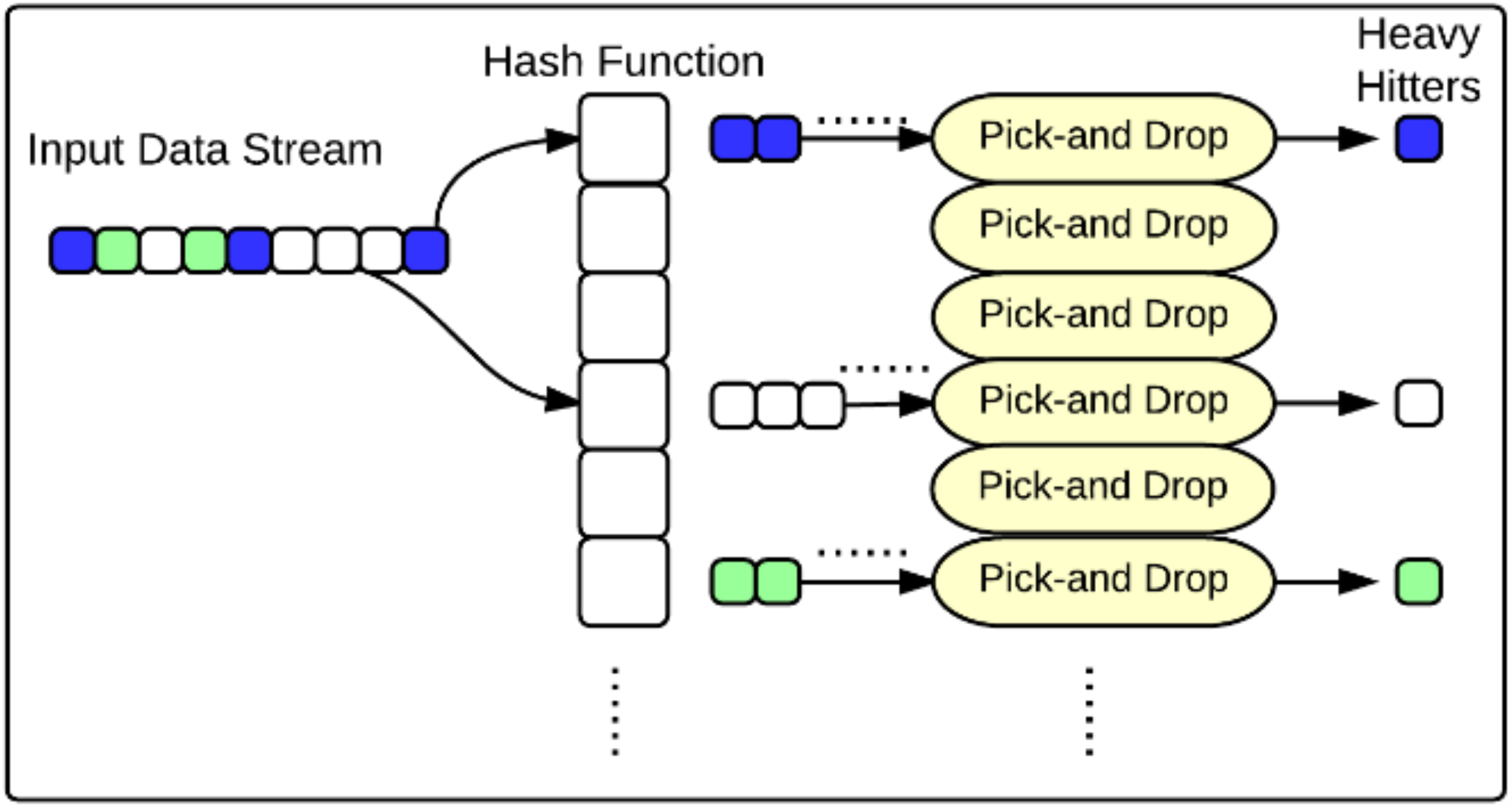
Pick-and-drop Algorithm:

- Sample a bunch of particles from the stream to approximate the heavy hitters.

Count Sketch

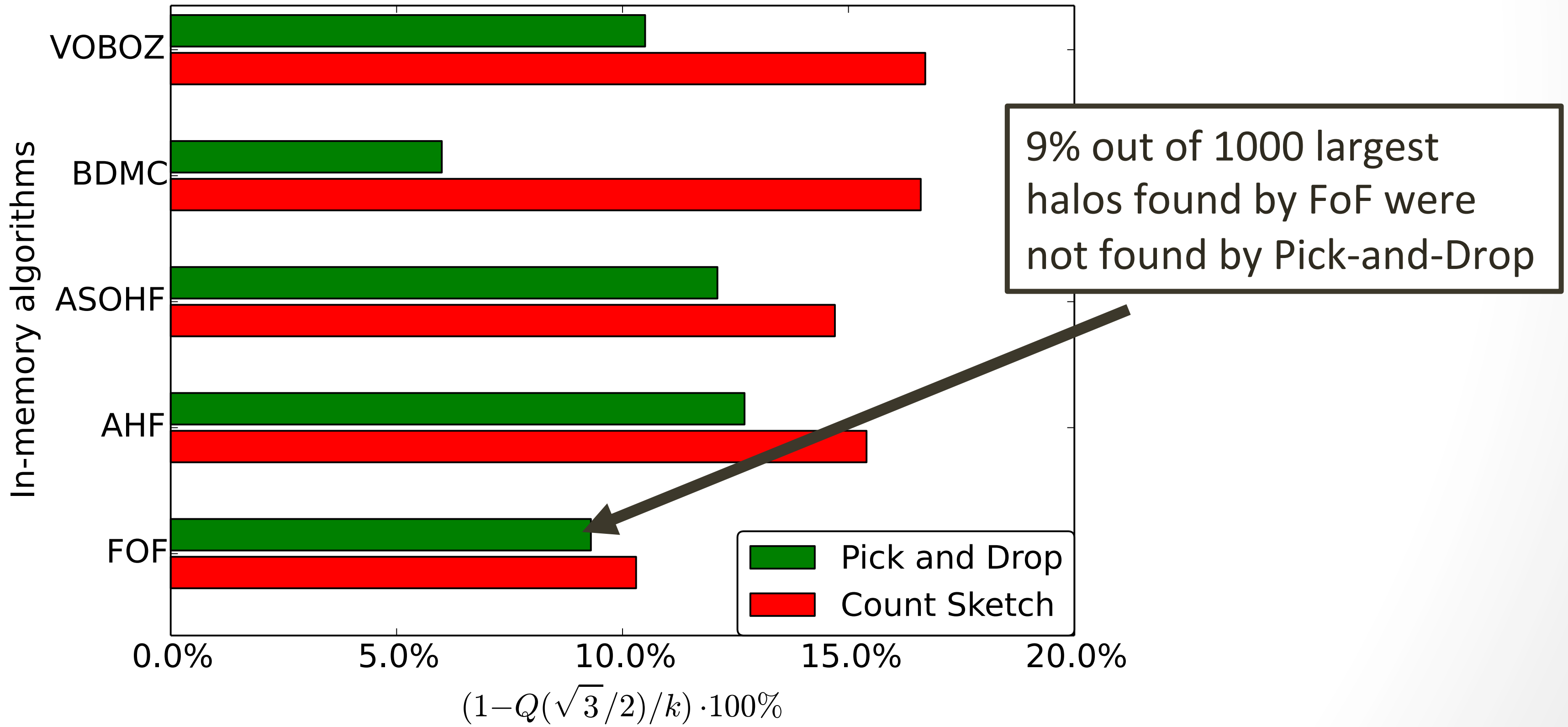


Pick and Drop



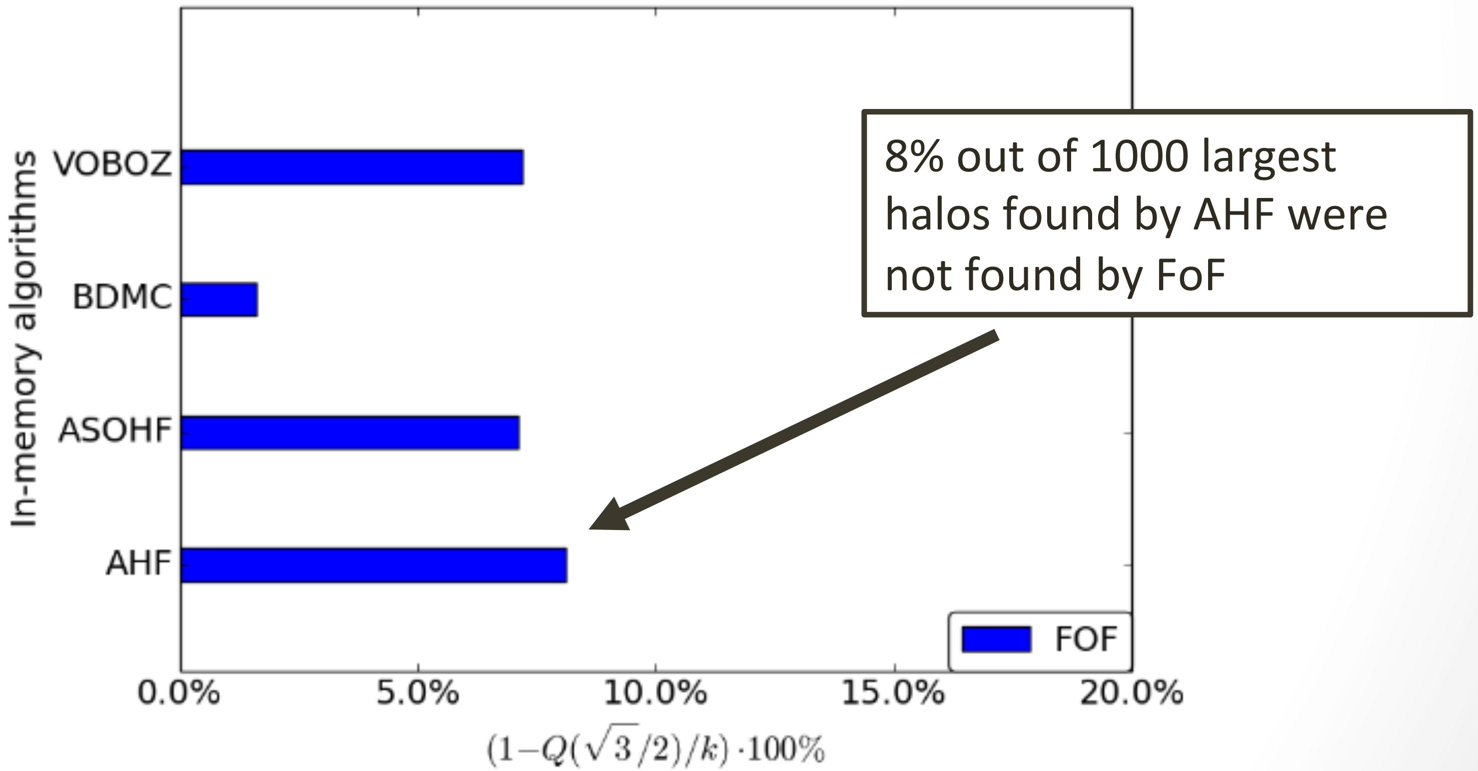
Evaluation

- Comparison with in-memory algorithms:
  - Percentage of haloes farther than a half-cell diagonal ($0.5\sqrt{3}$) from Pick-and-Drop and Count Sketch haloes



Evaluation

- Comparison of in-memory algorithms:
  - Percentage of haloes farther than a half-cell diagonal ($0.5\sqrt{3}$) from Friends-of-Friends haloes

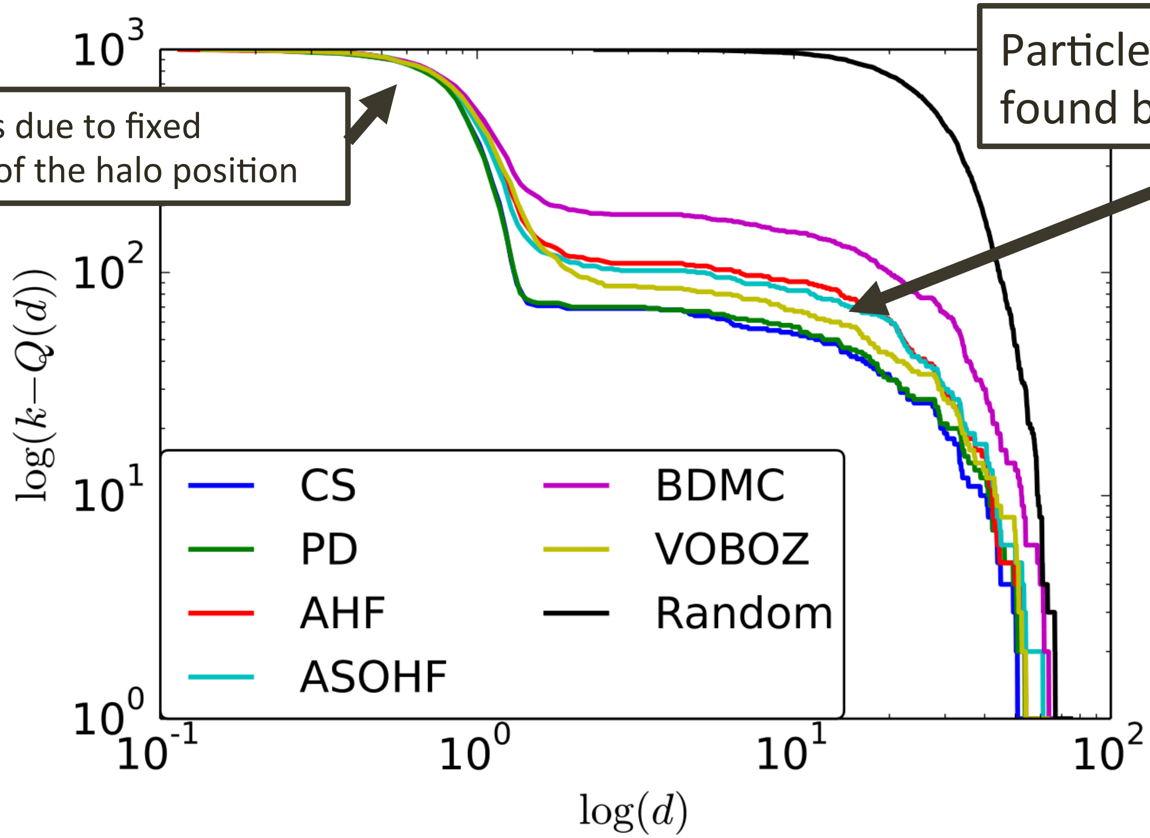


Evaluation

- Comparison with in-memory algorithms:
  - Number of top-1000 halos found by FOF farther than a distance d away from any of top-1000 halo from the algorithm of each curve

Randomness due to fixed assignment of the halo position

Particles that were not found by the algorithm



Memory

- Memory is the most significant advantage of applying streaming algorithms.

- Dataset size: $\sim 10\uparrow 9$ particles
  - Any in-memory algorithm: 12 GB
  - Pick-and-Drop: 30 MB

- GPU acceleration
  - One instance of Pick-and-Drop algorithm can be fully implemented by separate thread of GPU
  - Count Sketch algorithm have two time-consuming procedures: evaluating the hash functions and updating the queue. The first one can be naively ported to GPU

Summary

- We have provided connection between problem of halo finding and problem of heavy hitter search.
- Two streaming algorithms for finding top-k largest halos were compared with conventional halo finders.
- Low memory usage of these algorithm provide possibility to make computation on the laptop rather than huge computational cluster.
- Sublinearity of memory usage give us possibility to find top-k halos for much larger datasets in the future.

Future directions

- Develop algorithm that finds top-k largest halos for large k
- Investigate behavior of provided approaches in 6-dimensional space, where each particle represented by its position and velocity
- Modify algorithms so we can use extra information from spatial-friendly storing techniques

Thank you!

Support

This material is based upon work supported in part by the National Science Foundation under Grant No. 1447639, by the Google Faculty Award and by DARPA grant N660001-1-2-4014. Its contents are solely the responsibility of the authors and do not represent the official view of DARPA or the Department of Defense.

Complexities

- *Count Sketch:*

$O((k+F_2/f_k^2)\log n/\delta)$

- Pick-and-Drop:

$O(n^{1-2/K}\log n)$