

# Reframing Jet Physics with New Computational Methods

Kyle Cranmer, Matthew Drnevich,  
**Sebastian Macaluso**, Duccio Pappadopulo

vCHEP 2021

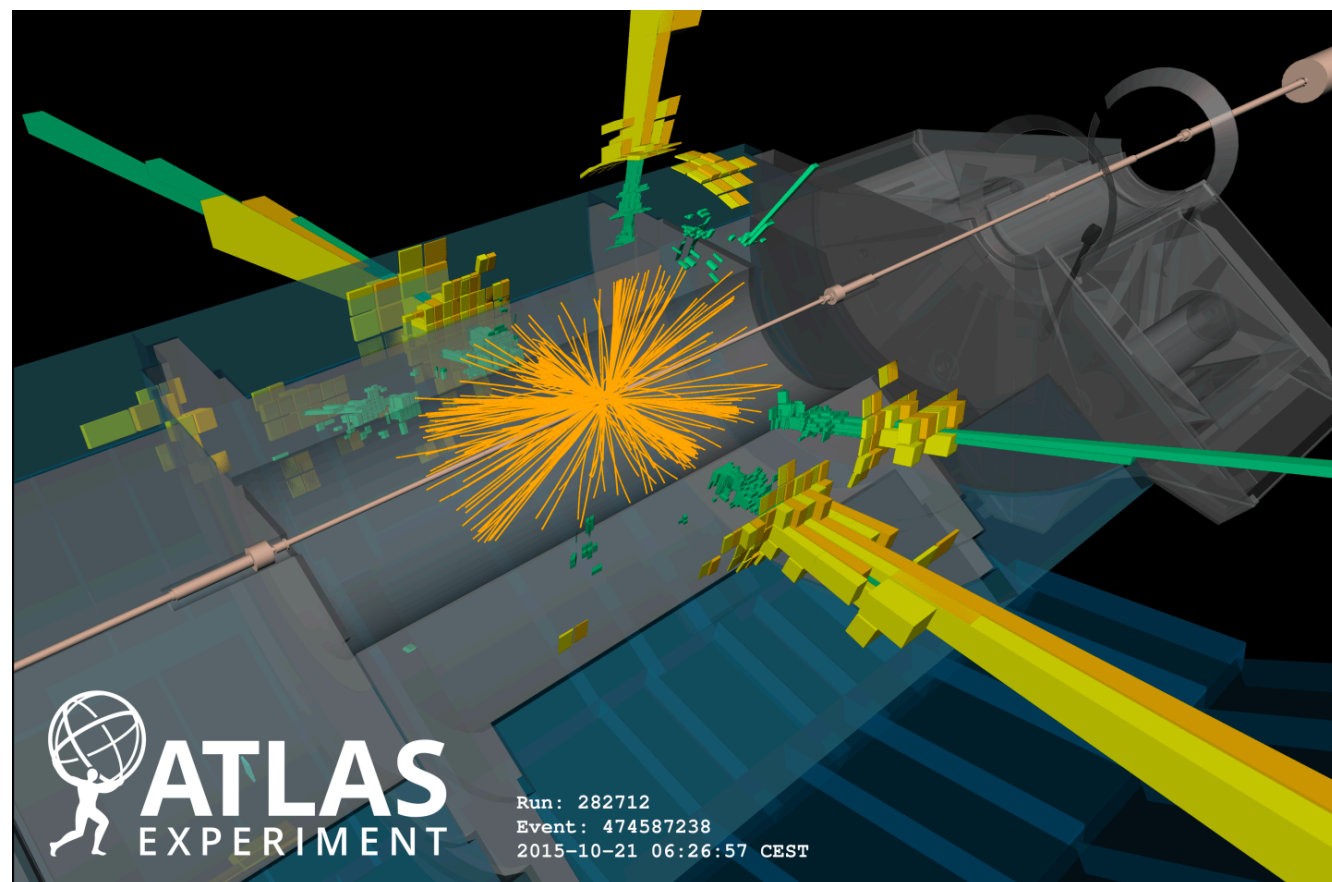
May 20, 2021



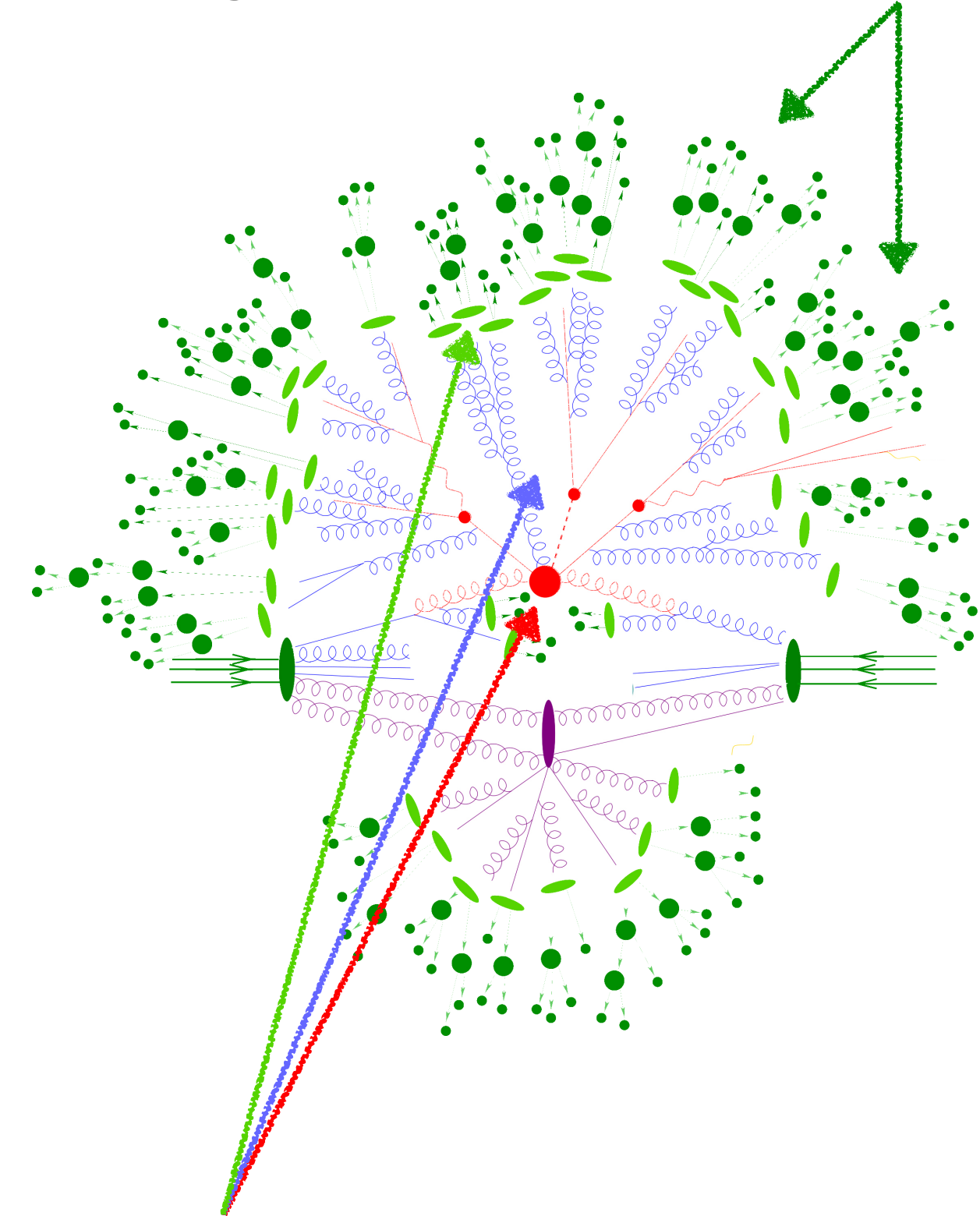
# Monte Carlo event generator

- Monte Carlo parton shower generators encode our understanding of the physics process that produces a jet.
- During simulation, successive splittings of the initial state particle generates a set of final state particles (leaves).
- Many showering histories (trees) could give rise to the same set of leaves.

To improve from first principles we need to think of jets in probabilistic terms.



We only **observe** the **particles**

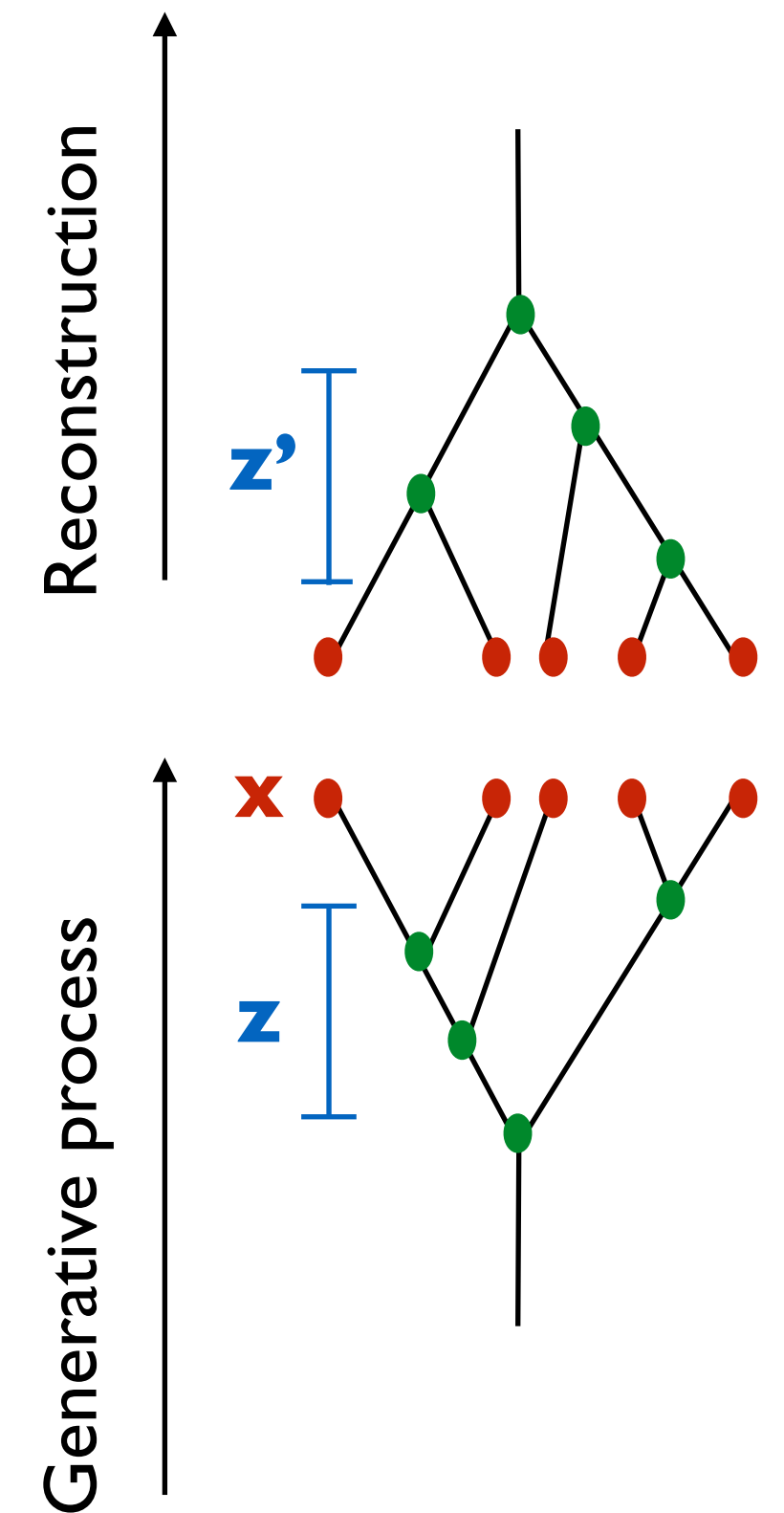


Evolution of the shower is **latent**

# Jet Reconstruction

- We want to invert the generative process.
- Task: reconstruct the unobserved showering history (latent tree)  $z$  from observed particles  $x$ .
- In more general terms we want to find the truth level hierarchical clustering given a set of leaves (jet constituents)  $x$ .

Hierarchical Clustering  
Recursive partitioning of a data set into successively smaller clusters.



# Generalized kt clustering algorithms

- Idea: sequentially cluster jet constituents aiming to recover the showering history.
- Bottom-up (agglomerative) clustering.
- Merge closest pair based on a distance measure.
- Intuitively, particles with smaller  $d_{ij}$  have a greater likelihood to have come from a common parent.

$$d_{ij}^{(\alpha)} = \min(p_{ti}^{2\alpha}, p_{tj}^{2\alpha}) \frac{\Delta R_{ij}^2}{R^2}$$

$$\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$$

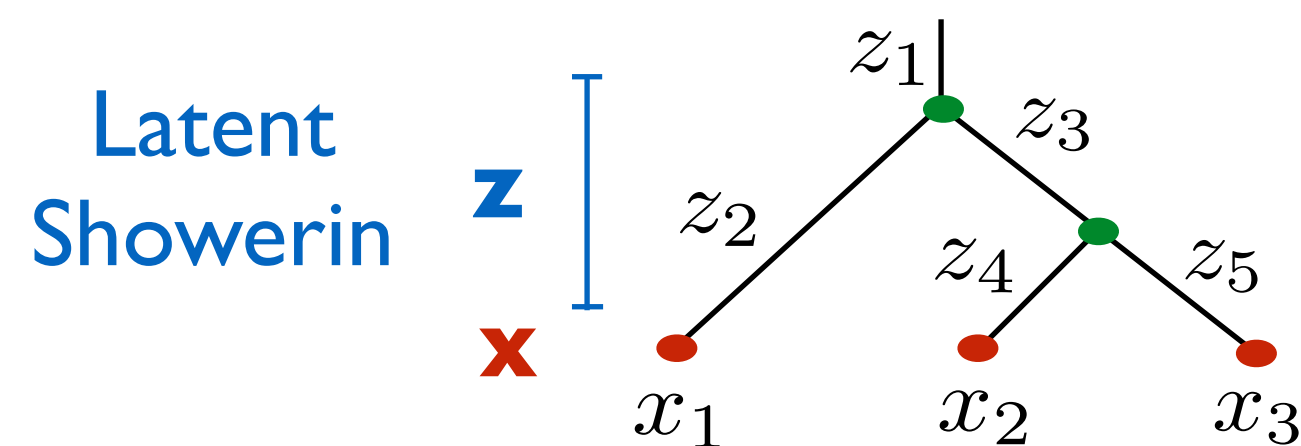
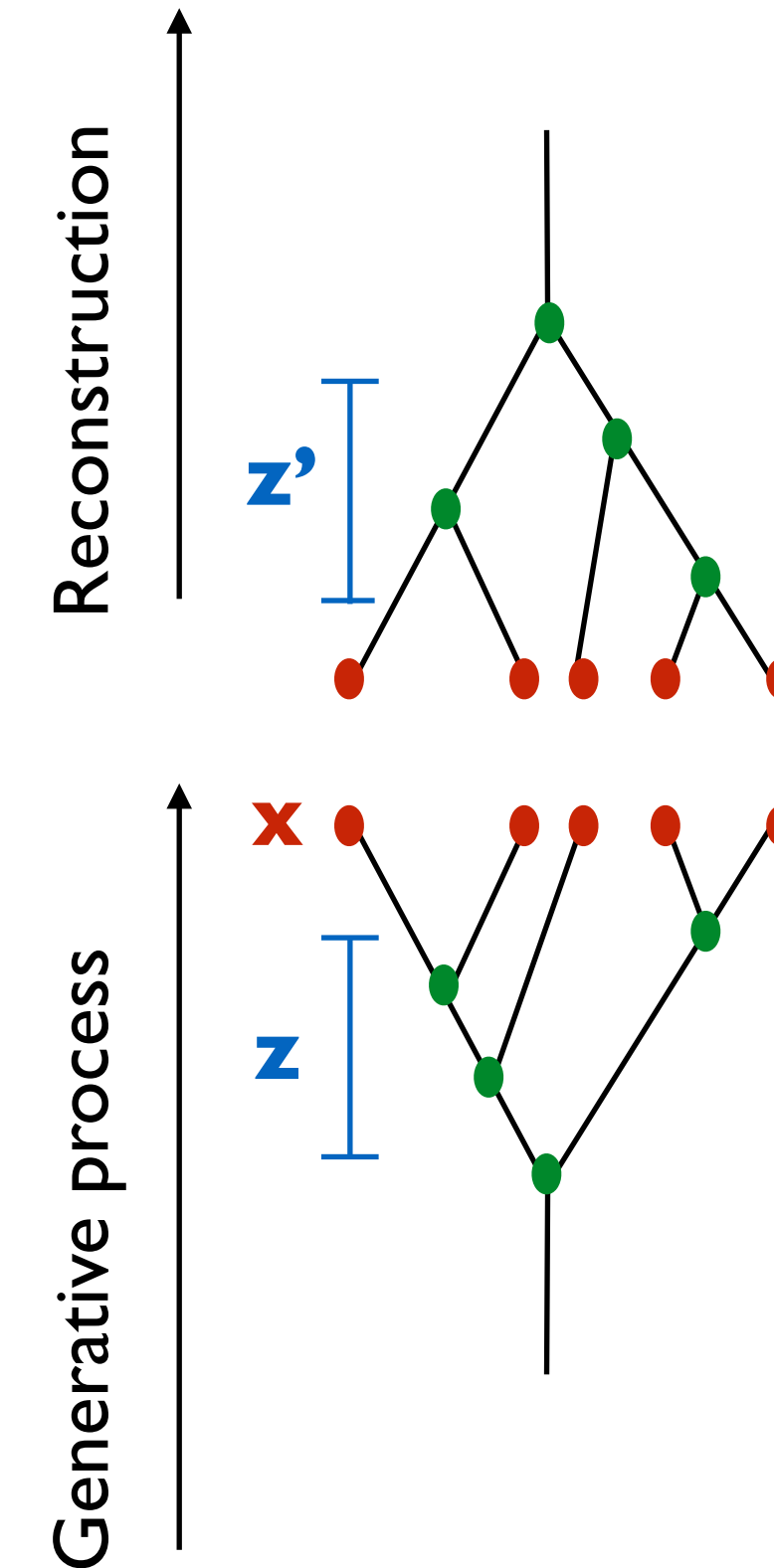
$\alpha = \{1, 0, -1\}$  specifies the the kt, Cambridge/Aachen and anti-kt algorithms.

kt-like clustering algorithms use a heuristic, with no explicit connection to the generative model.

<p><b>The anti-<math>k_t</math> jet clustering algorithm</b> #1</p> <p>Matteo Cacciari (Paris, LPTHE), Gavin P. Salam (Paris, LPTHE), Gregory Soyez (Brookhaven) (Feb, 2008)</p> <p>Published in: <i>JHEP</i> 04 (2008) 063 • e-Print: 0802.1189 [hep-ph]</p> <p>pdf DOI cite</p>	<p>7,689 citations</p>
<p><b>FastJet User Manual</b> #1</p> <p>Matteo Cacciari (Paris, LPTHE and Diderot U., Paris), Gavin P. Salam (CERN and Princeton U. and Paris, LPTHE), Gregory Soyez (Saclay, SPHT) (Nov, 2011)</p> <p>Published in: <i>Eur.Phys.J.C</i> 72 (2012) 1896 • e-Print: 1111.6097 [hep-ph]</p> <p>pdf DOI cite</p>	<p>3,954 citations</p>

# Reframing jet physics in probabilistic terms

- Joint likelihood:  $p(x, z|\theta)$
- Marginal likelihood:  $p(x|\theta) = \int dz p(x, z|\theta)$
- Maximum likelihood history:  $\text{ArgMax}_z p(x, z|\theta)$
- Posterior distribution on histories:  $p(z|x, \theta)$
- Maximum likelihood parameter:  $\text{ArgMax}_\theta p(x|\theta)$
- Posterior distribution on  $\theta$  :  $p(\theta|x)$



Joint likelihood:

$$p(x, z|\theta) = \prod_j p(x_j | z_{\text{parent}(x_j)}, \theta) \prod_i p(z_i | z_{\text{parent}(z_i)}, \theta)$$

# Parton Showers generators

PYTHIA



Sherpa

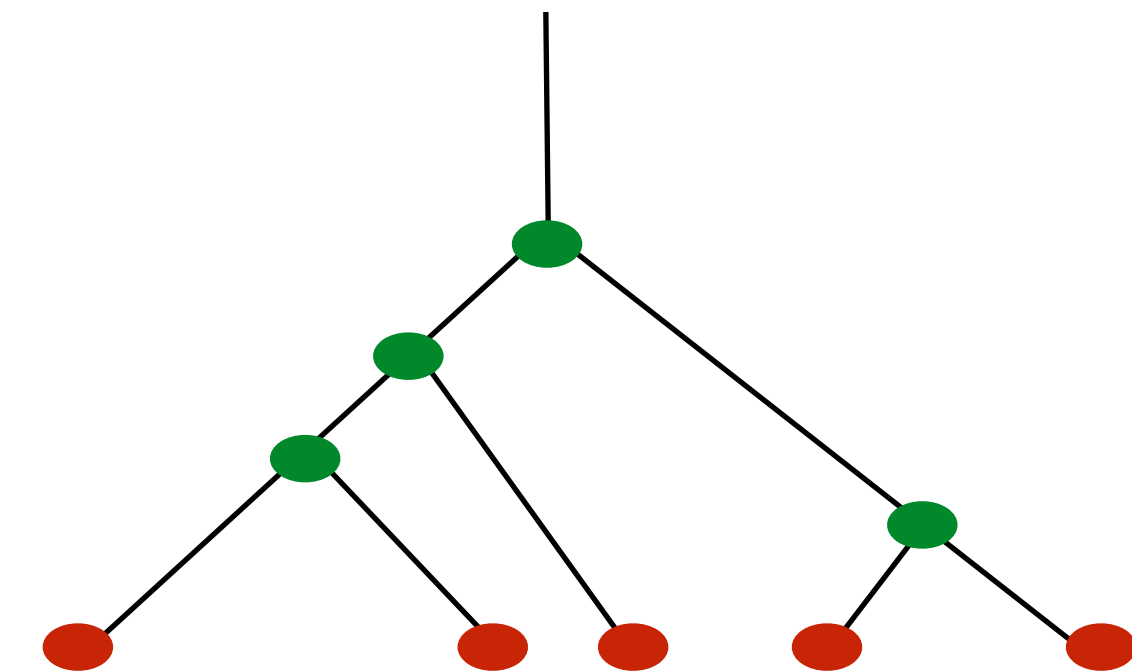


Herwig 7

- It is hard to access the joint likelihood of a showering process.
- Other complications related to momentum conservation that could be addressed [Bauer & Tackmann '08].

$$\mathcal{P}_{\text{PS}}(\{t_i, z_i\}) = \mathcal{P}(t_1, z_1) \times \mathcal{P}(t_2, z_2) \times \dots$$

To prototype we built our own simplified model!



# Ginkgo: Simplified Generative Model for Jets

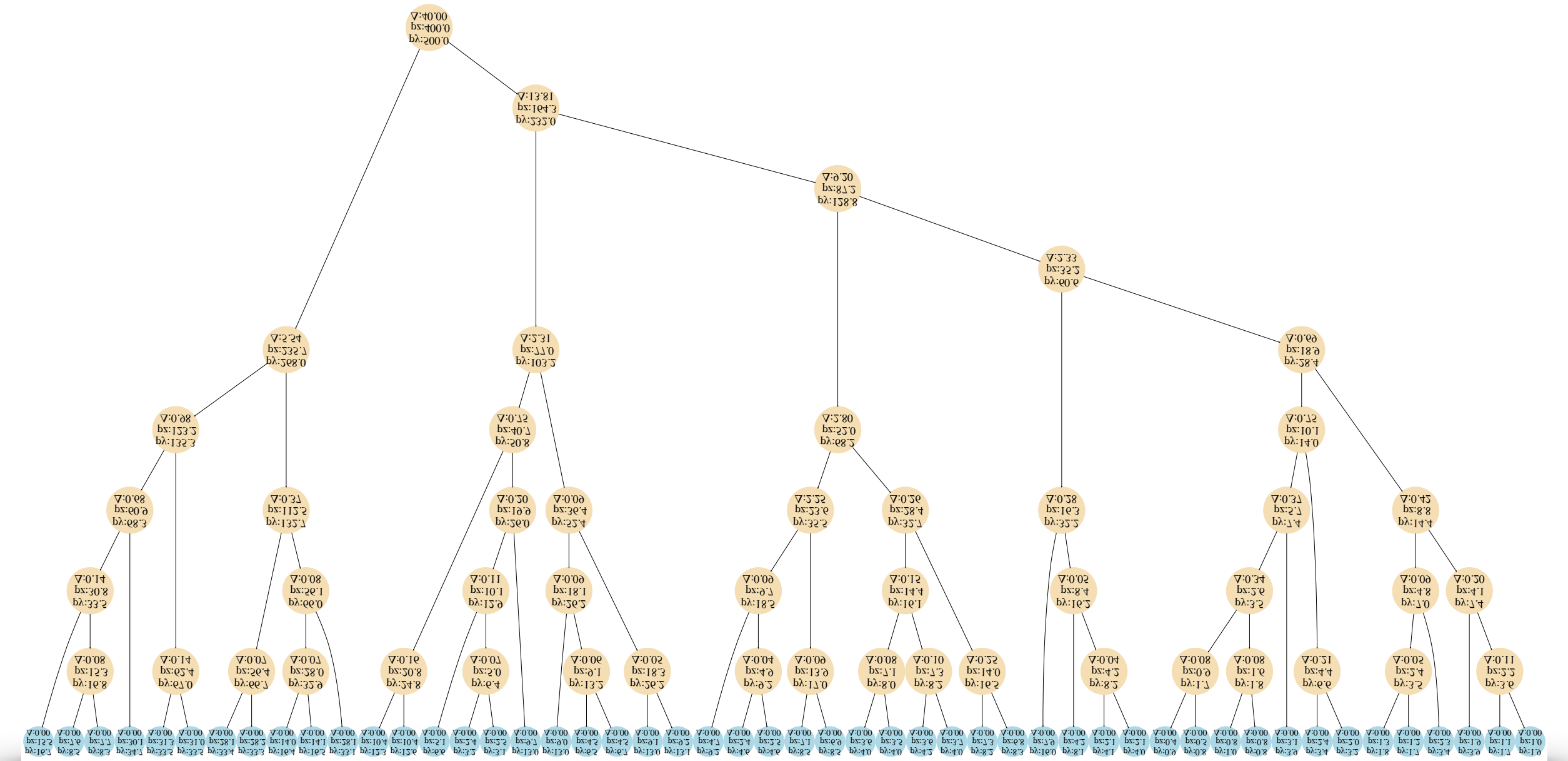


Kyle Cranmer, SM &  
Duccio Pappadopulo

[github.com/  
SebastianMacaluso/  
ToyJetsShower](https://github.com/SebastianMacaluso/ToyJetsShower)

## Motivation

- Build a model to aid in ML research for jet physics.
- Facilitate exact/approximate combinatorial optimization.
- Marginalize over showering histories (binary trees).



## Generation

- Tractable joint likelihood.
- Captures essential ingredients of parton shower generators in full physics simulations.
- Implements an analogue to a parton shower (no hadronization effects).
- Python implementation with few software dependencies.

## Inference

- E.g. tuning of simulation parameters (PYTHIA TUNES) to optimize a fit to the data.

# Model description

Recursive algorithm to generate a binary tree with jet constituents as the leaves.

Showering process: binary splittings + stopping rule.

## Features

- Momentum conservation.
- Running of the splitting scale.

## Facilitates research with:

- Probabilistic programming
- Differentiable programming
- Dynamic programming
- Variational inference

---

### Algorithm 1: Toy Parton Shower Generator

---

1 function NodeProcessing ( $p_p^\mu, t_P, t_{\text{cut}}, \lambda, \text{tree}$ )

**Input** : parent momentum  $\vec{p}_p$ , parent mass squared  $t_P$ , cut-off mass squared  $t_{\text{cut}}$ , rate for the exponential distribution  $\lambda$ , binary tree  $\text{tree}$

2     Add parent node to tree.

3     **if**  $t_P > t_{\text{cut}}$  **then**

4         Sample  $t_L$  and  $t_R$  from the decaying exponential distribution.

5         Sample a unit vector from a uniform distribution over the 2-sphere.

6         Compute the 2-body decay of the parent node in the parent rest frame.

7         Apply a Lorentz boost to the lab frame to each child.

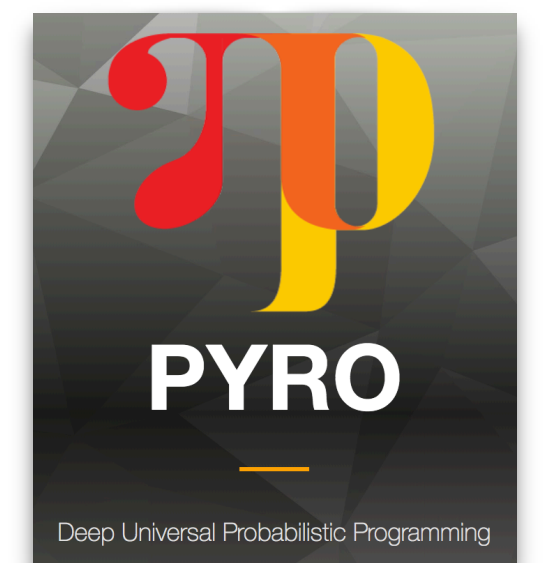
8         NodeProcessing ( $p_p^\mu, t_L, t_{\text{cut}}, \lambda, \text{tree}$ )

9         NodeProcessing ( $p_p^\mu, t_R, t_{\text{cut}}, \lambda, \text{tree}$ )

---

$$t_L \sim f(t|\lambda, t_P) = \frac{1}{1 - e^{-\lambda t_P}} \frac{\lambda}{t_P} e^{-\frac{\lambda}{t_P} t} \quad t_R \sim f(t|\lambda, t_P, t_L) = \frac{1}{1 - e^{-\lambda t_P}} \frac{\lambda}{(\sqrt{t_P} - \sqrt{t_L})^2} e^{-\frac{\lambda}{(\sqrt{t_P} - \sqrt{t_L})^2} t}$$

The model keeps track of the augmented data based on a **PYRO** implementation.



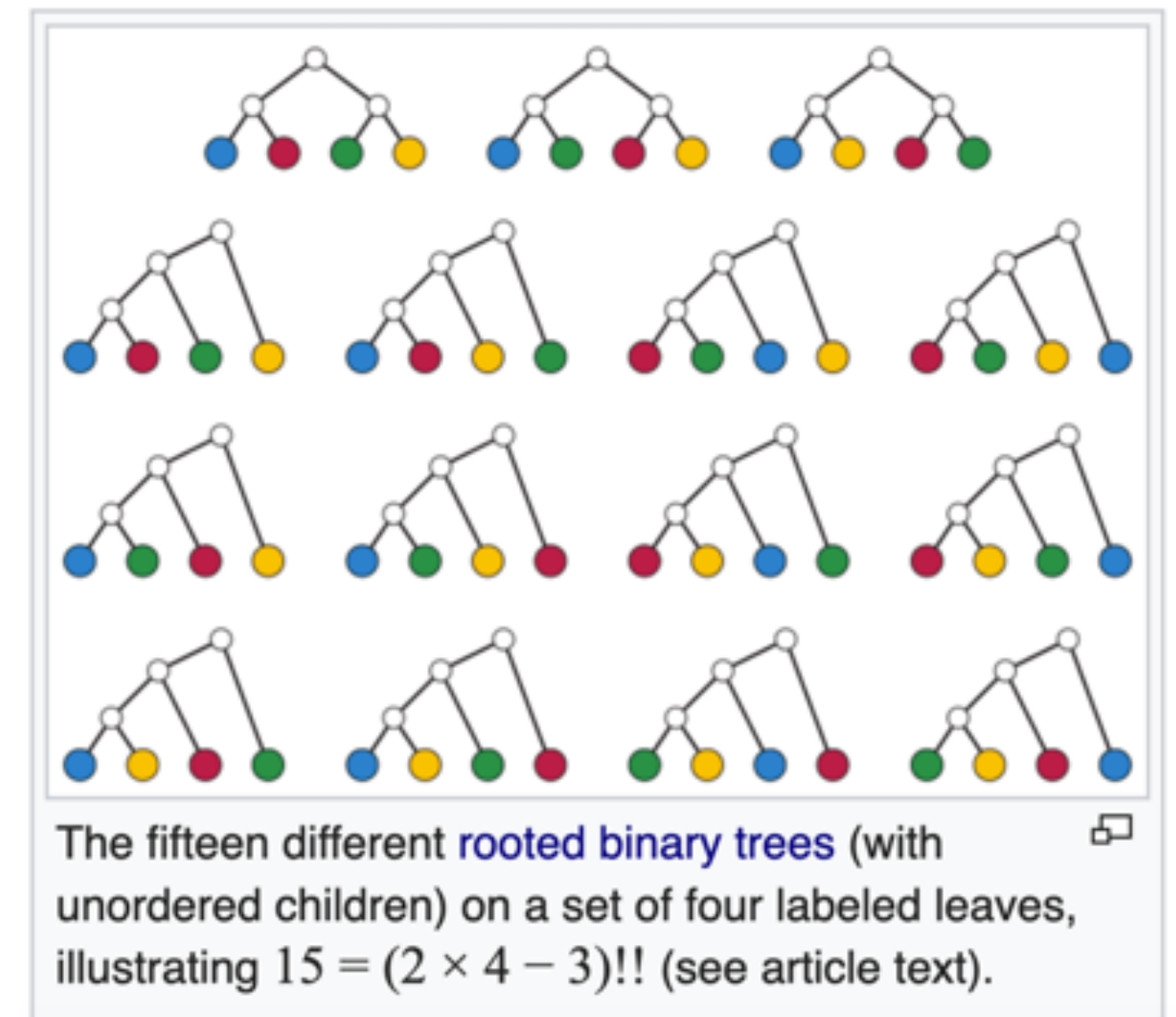


# Generation and inference unification

We aim to invert the process probabilistically

- Reconstructing the showering history is like inverting the generative model.
- The number of clustering histories is enormous! It grows like  $(2N-3)!!$ , with  $N$  the number of jet constituents.
- Traditional jet reconstruction algorithms don't use the probability model directly.
- We use the likelihood of the showering history as the optimization objective.

# of leaves	Approx. # of trees
4	15
5	100
7	10 K
9	2 M
11	600 M
150	$10^{300}$



[https://en.wikipedia.org/wiki/Double\\_factorial](https://en.wikipedia.org/wiki/Double_factorial)

# Improving over sequential (agglomerative) clustering

<https://github.com/SebastianMacaluso/StandardHC>

- In the probabilistic language, generalized kt algorithms are greedy.

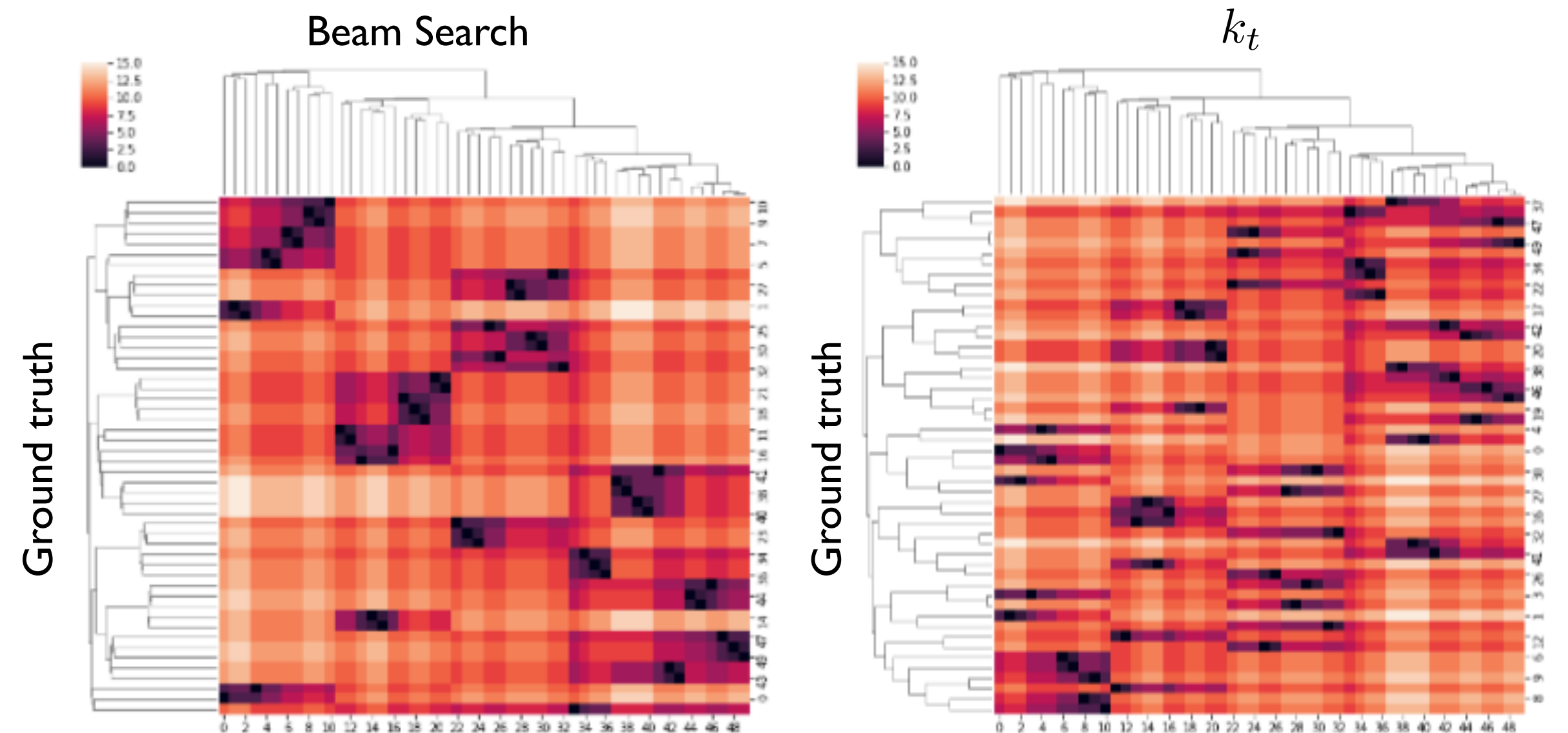
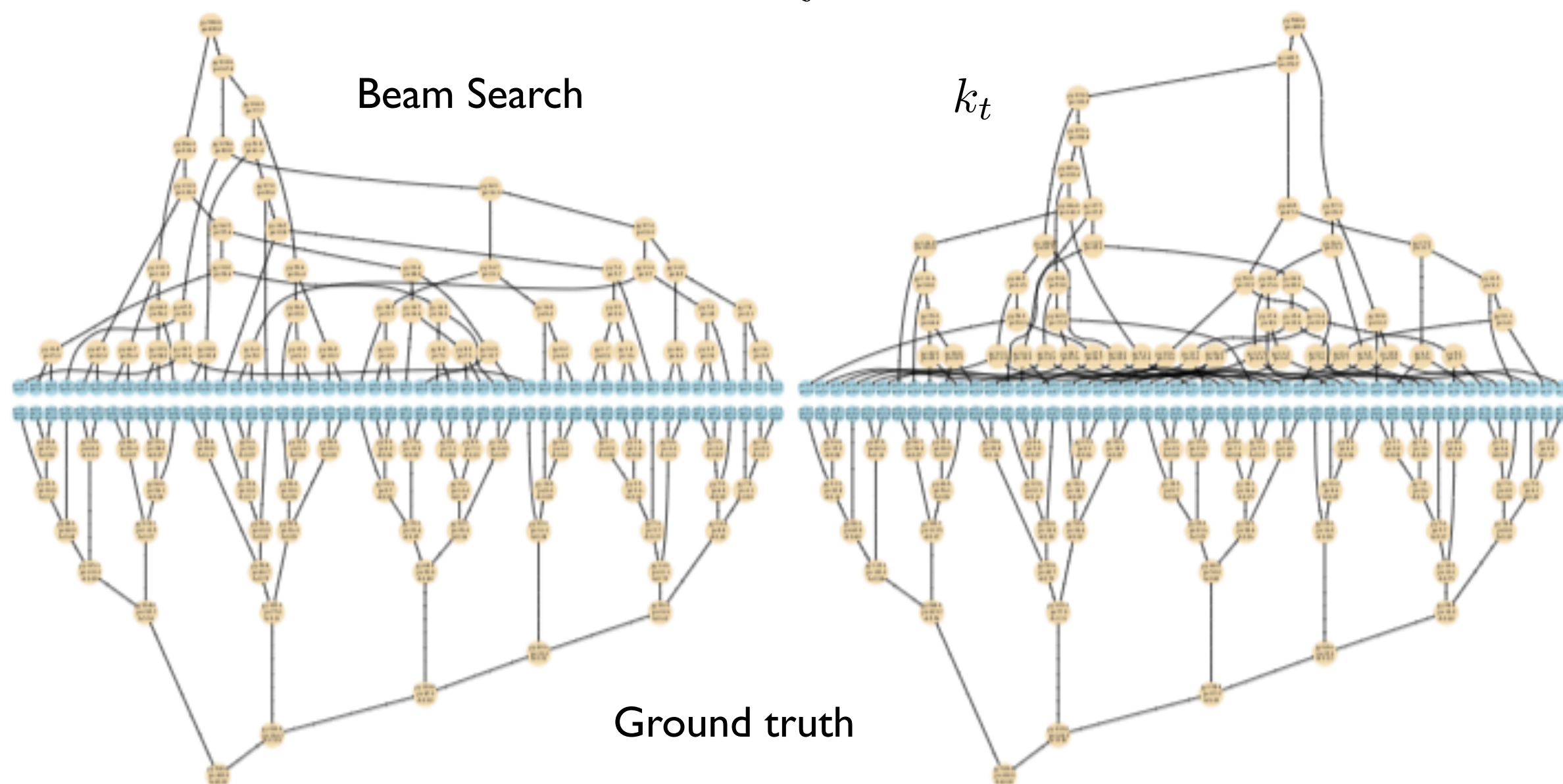
Greedy algorithms are analogue to playing chess only thinking one move at a time.

Straightforward improvements:

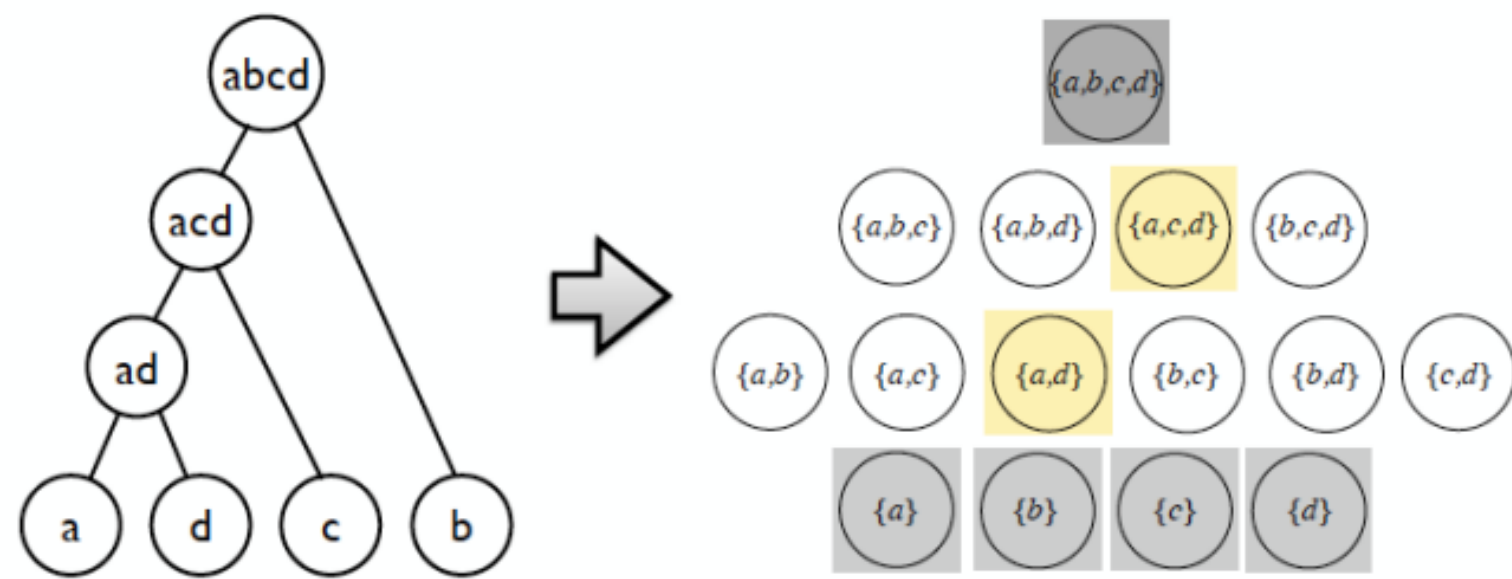
- Use the splitting likelihood encoded in the generative model instead of heuristics.
- Splitting likelihood gives a natural way to score the combination of multiple clusterings, i.e. their product. This allows to explore other algorithms, e.g. beam search.

Beam Search: keeps multiple possible clusterings in memory before choosing the showering history.

$$\hat{z} = \operatorname{argmax}_z p(z | x, \theta)$$



- New data structure and dynamic programming algorithms to efficiently consider every possible showering history (hierarchical clustering).
  - Exact Maximum likelihood showering history  $\text{ArgMax}_z p(x, z|\theta)$
  - Marginal likelihood  $p(x|\theta)$  (sum of the likelihood of all the showering histories).



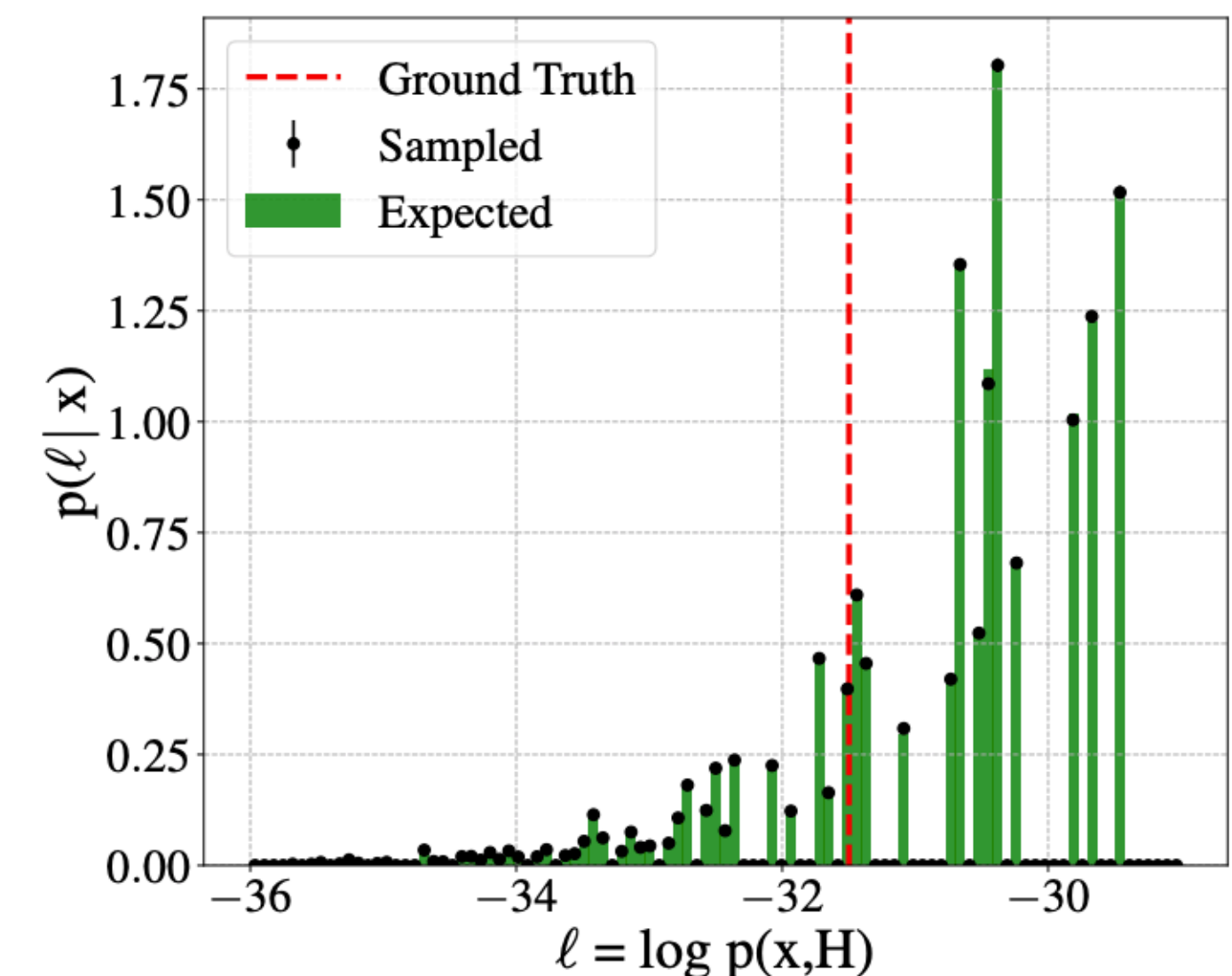
We connected with Andrew McCallum's group at UMass



Craig Greenberg    Nicholas Monath

## Posterior distribution - Sampling procedure

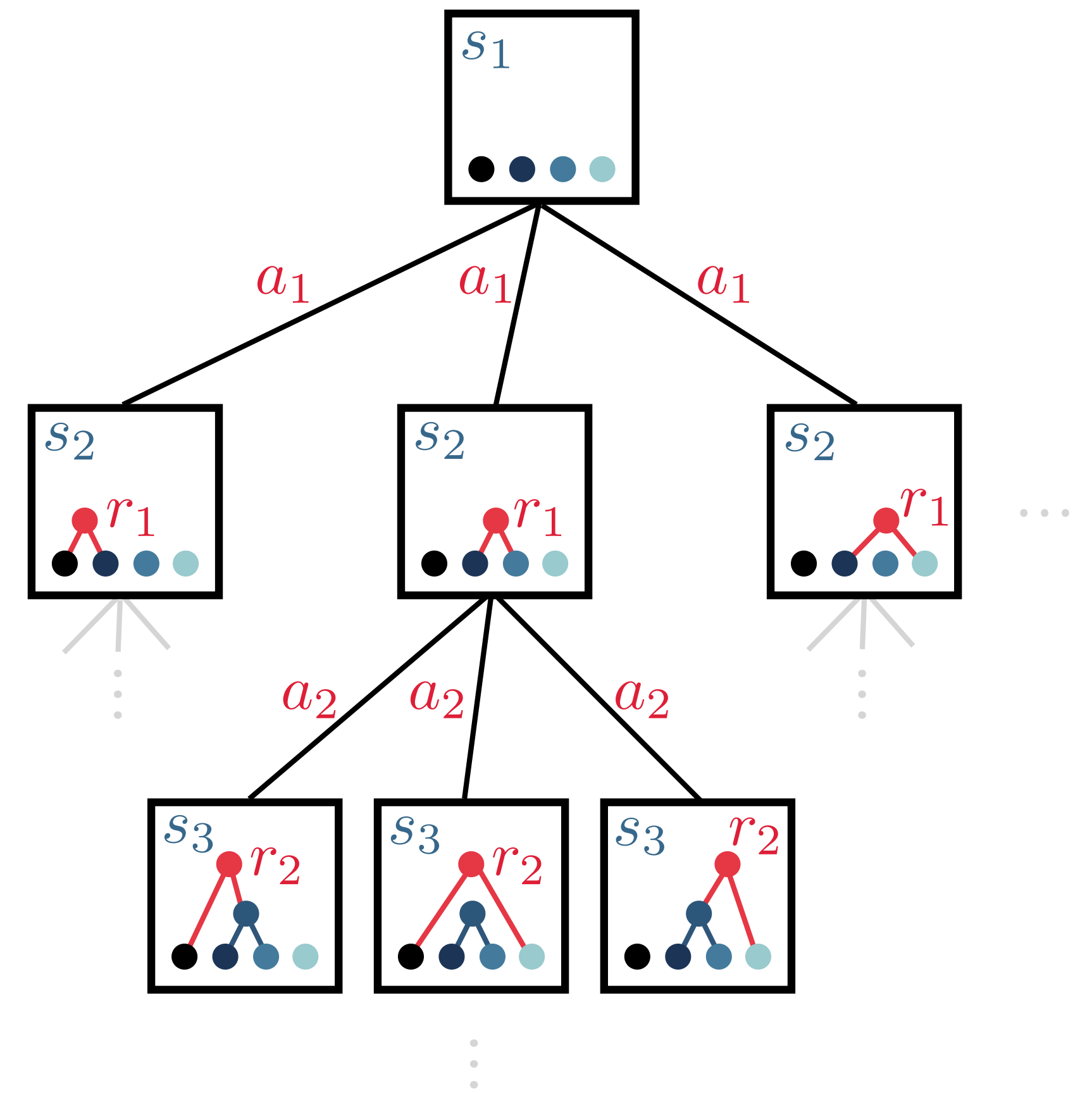
- Trellis enables to sample histories weighted by their likelihood from the true posterior distribution.
  - Useful to implement the CKKW-L matching algorithm for events with large jet multiplicity.



# Hierarchical Clustering as a Markov Decision Process

Brehmer, Macaluso, Pappadopulo,  
Cranmer [arXiv:2011.08191]  
NeurIPS 2020 ML4Physical Sciences  
workshop

- The state space  $\mathbf{S}$  is given by all possible particle sets at any given point during the clustering process.
- The actions  $\mathbf{A}$  are the choice of two particles to be merged.
- The state transitions  $\mathbf{P}$  are deterministic and update  $z_t$  to  $z_{t-1}$
- The rewards  $\mathbf{R}$  are the splitting probabilities.
- The MDP is episodic and terminates when only a single particle is left.



We implement Monte Carlo Tree Search (MCTS) and Imitation learning / Behavioral Cloning (BC).

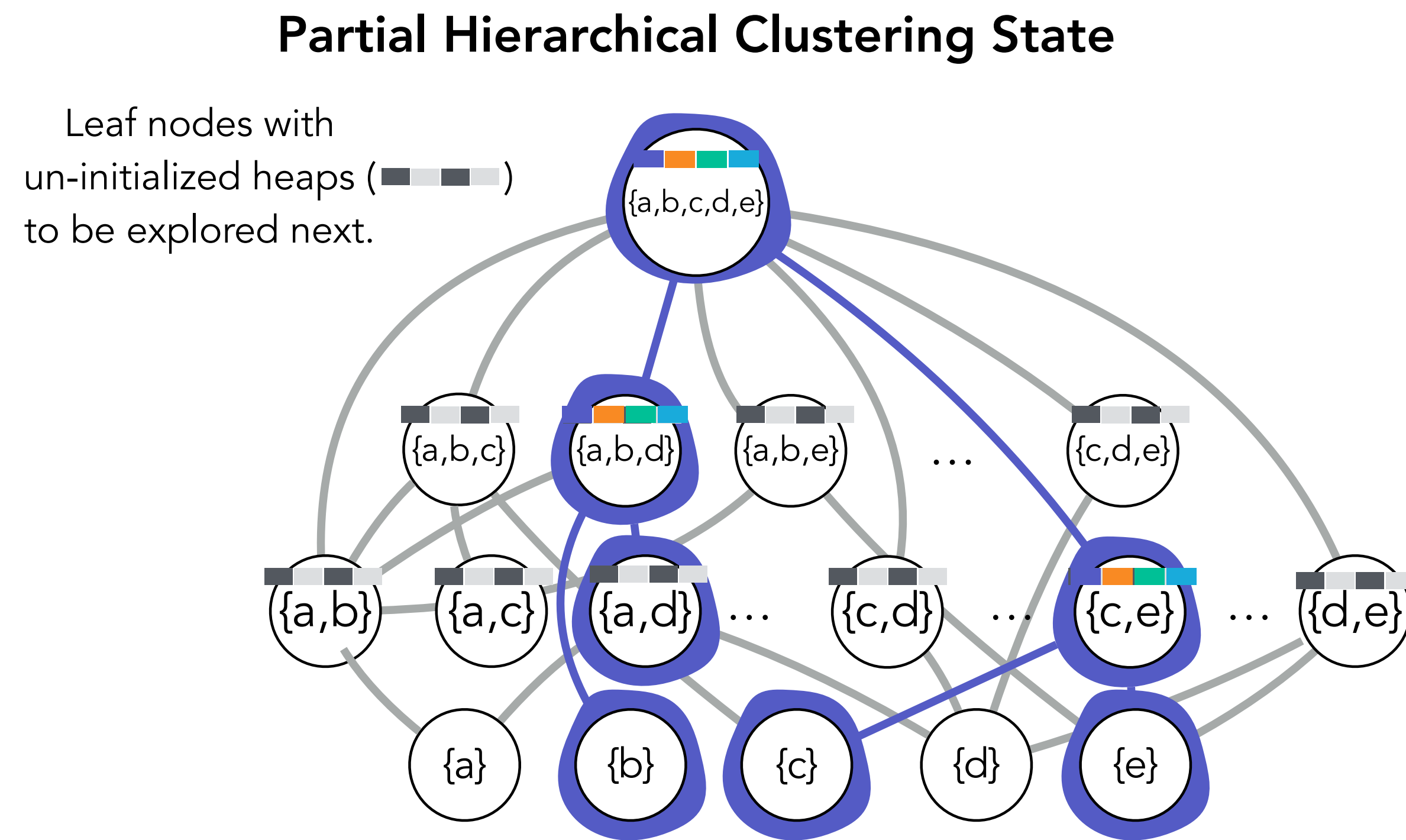
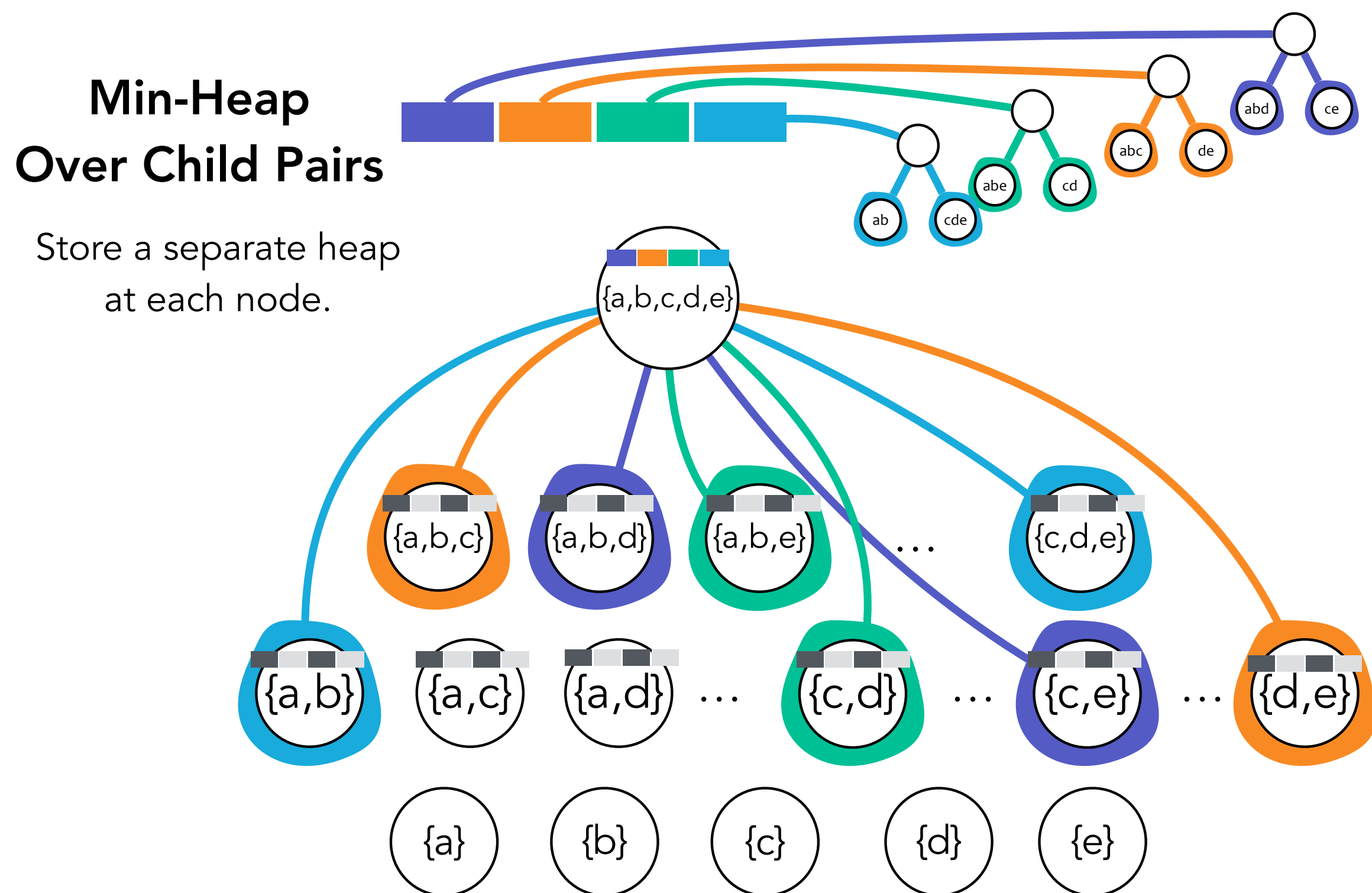
- We proposed A\* search on the trellis to find the MLE hierarchy.
- Best-first search algorithm that redefines clustering as search.
- Based on a heuristic to determine the most promising path.
- Trellis compactly encodes states in the search space and allows a top-down exploration.
- Compactly represents the search frontier as nested priority queues.
- Approximate version based on a sparse trellis and/or limiting the number of trees explored.



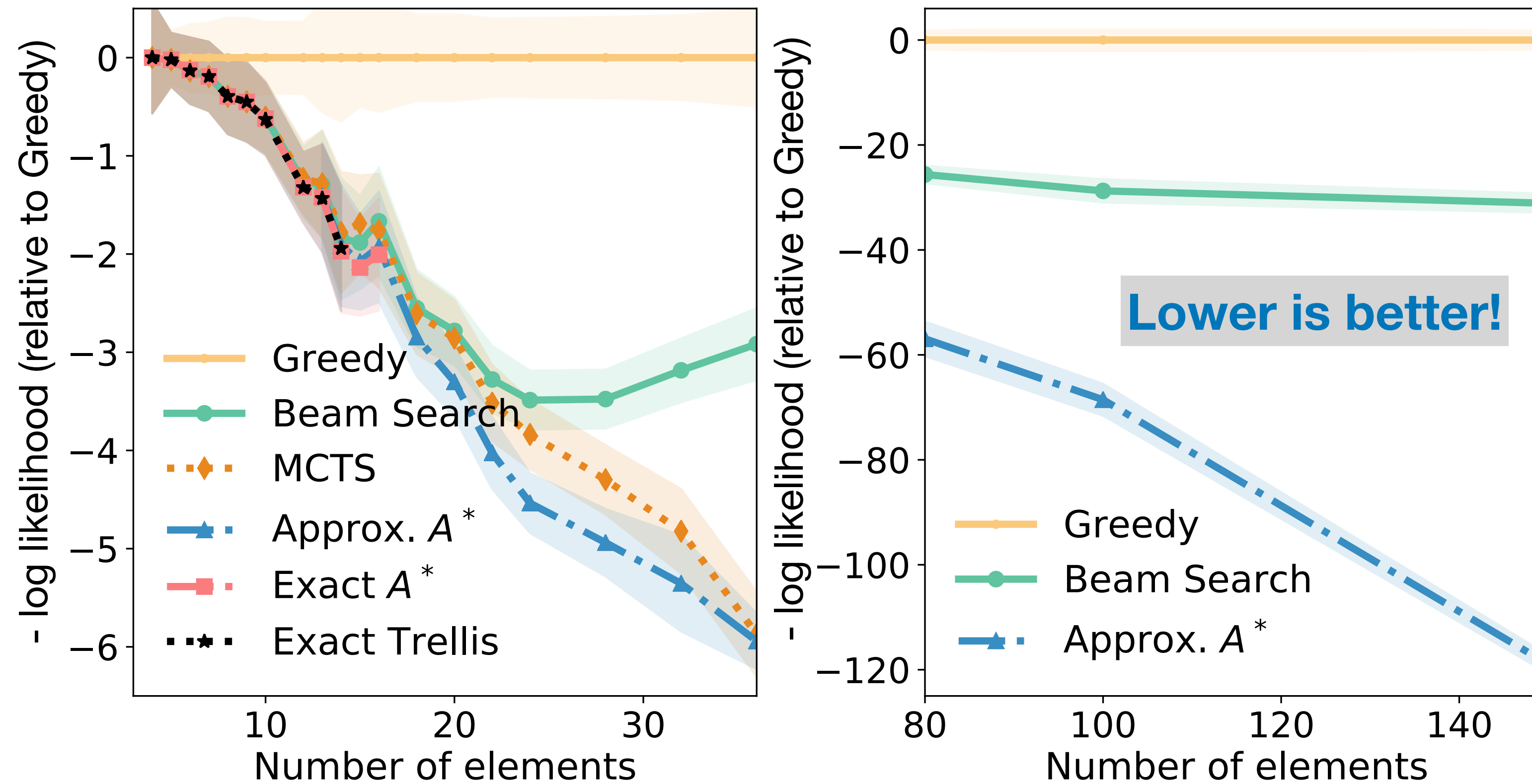
Craig Greenberg



Nicholas Monath



- We find the exact maximum likelihood tree for up to 16 jet constituents.
- Our approx. algorithms greatly improve over greedy and beam search baselines.

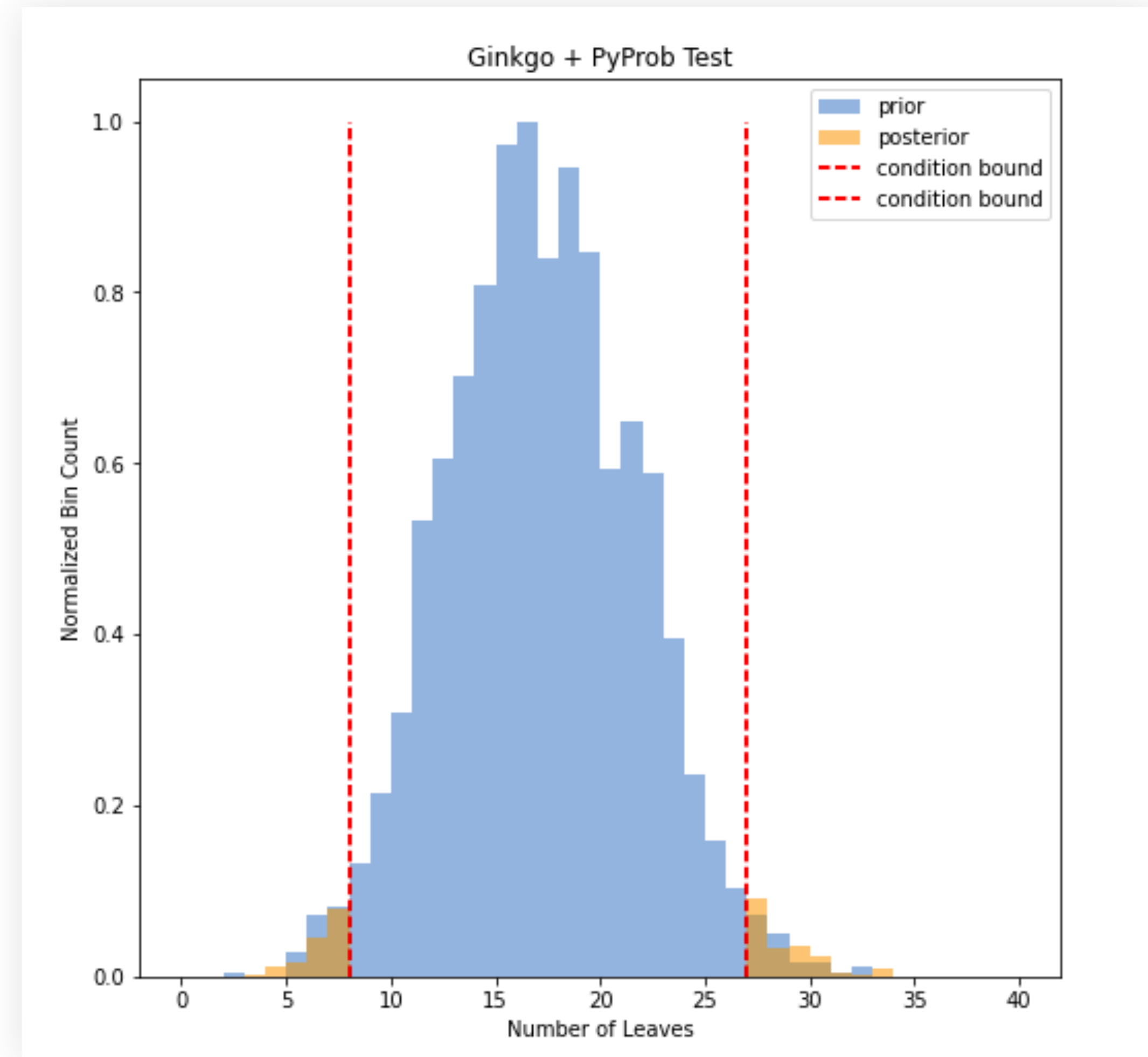


# of leaves	Approx. # of trees
4	15
5	100
7	10 K
9	2 M
11	600 M
150	$10^{300}$

- Generating sufficient background in signal-like tails is hard, e.g. background for boosted jet taggers.
- Monte Carlo parton showers sample jets through random number generators.
- Probabilistic programming offers a way to hijack the random number generators and sample from this complicated region of phase space, e.g. importance sampling.
- Applied to Sherpa

NeurIPS 2019 [Baydin, Heinrich, Louppe, Cranmer, et al, arXiv:1807.07706]  
SC 2019 [Baydin, Louppe, Cranmer et al, arXiv:1907.03382]

Allows to efficiently sample the tails of backgrounds in signal-rich regions of phase space.



Importance sampling on Ginkgo jets using PyProb and conditioning on the number of constituents.

# Final remarks

---

- Reframed jet physics in probabilistic terms.
- Introduced a simplified generative model to facilitate research into new computational techniques for jet physics.
- Reframed jet clustering as a way to invert the generative model.
- New implementations of likelihood-based clustering algorithms: greedy and beam search that provide a principled alternative to the generalized  $k_t$  clustering algorithms.
- Hierarchical cluster trellis and  $A^*$  algorithms to exactly obtain the maximum likelihood showering history (approximate versions greatly improve over baselines).
- New implementations of reinforcement learning based algorithms such as MCTS for jet clustering.
- Cluster trellis allows to marginalize and sample from the true posterior distribution of showering histories which could ameliorate bottlenecks in physics simulations with large jet multiplicity.
- Probabilistic programming allows to efficiently sample the tails of backgrounds in signal-rich regions of phase space.

**Thanks for your attention!**



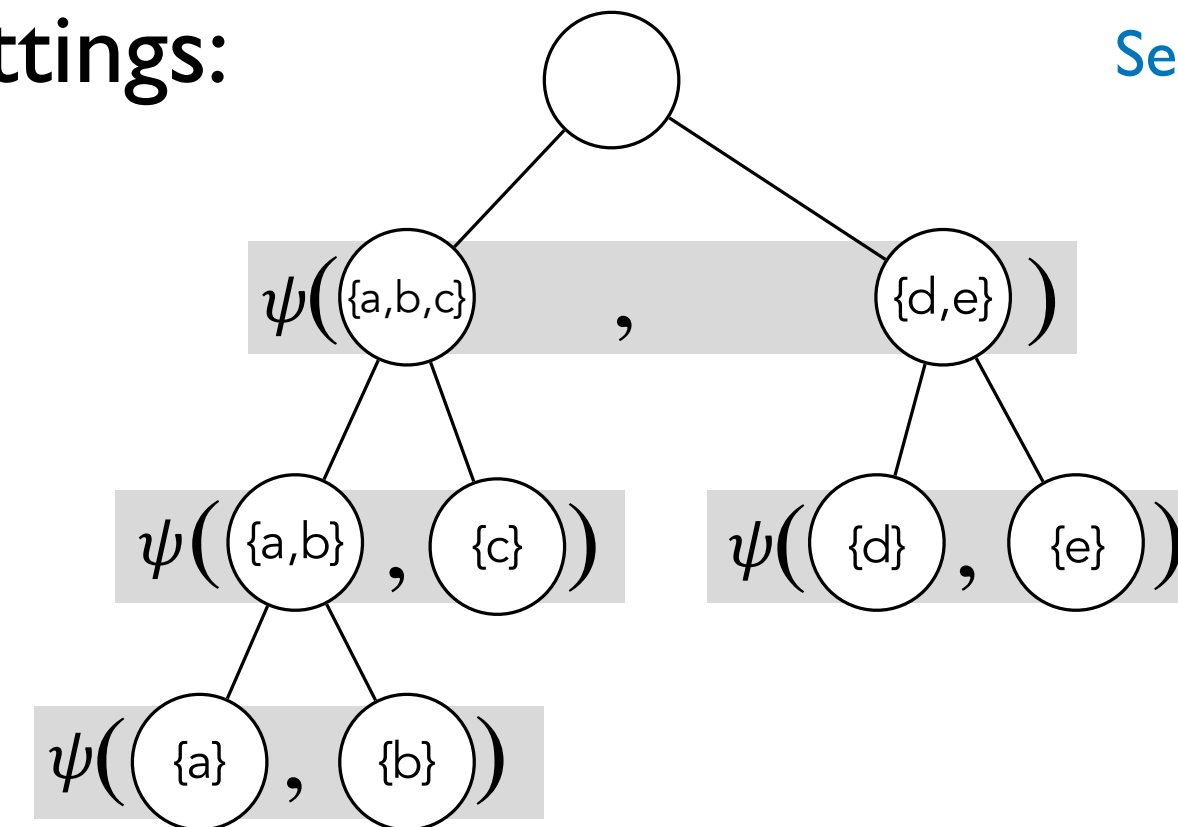




- New data structure to efficiently consider every possible hierarchical clustering.
  - Exact Maximum likelihood showering history  $\text{ArgMax}_z p(x, z|\theta)$
  - Marginal likelihood  $p(x|\theta)$  (sum of the likelihood of all the clustering histories).
- Model needs to be defined in terms of the product of pairwise splittings:

$$P(H|X) = \frac{\phi(X|H)}{Z(X)} \text{ with } \phi(X|H) = \prod_{X_L, X_R \in \text{sibs}(H)} \psi(X_L, X_R)$$

$$Z(X) = \sum_{H \in \mathcal{H}(X)} \phi(X|H). \quad \text{Marginal likelihood}$$



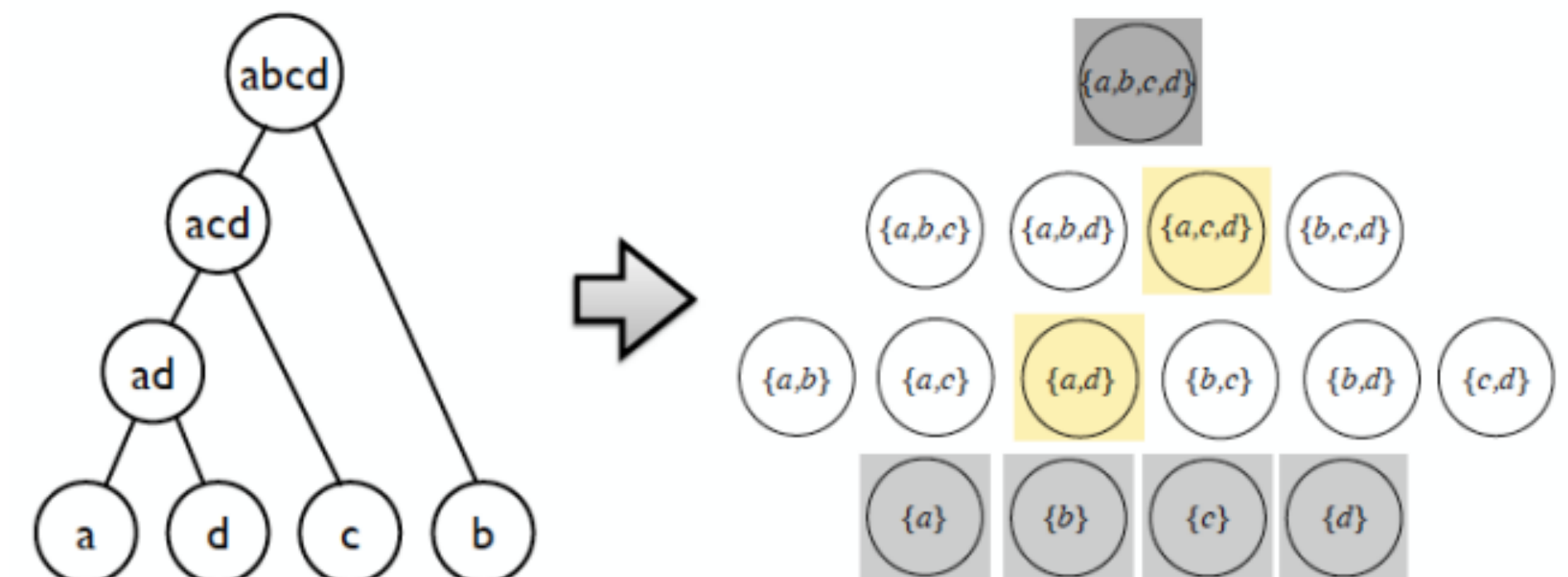
We connected with Andrew McCallum's group at UMass



Craig Greenberg    Nicholas Monath

<https://github.com/SebastianMacaluso/ClusterTrellis>

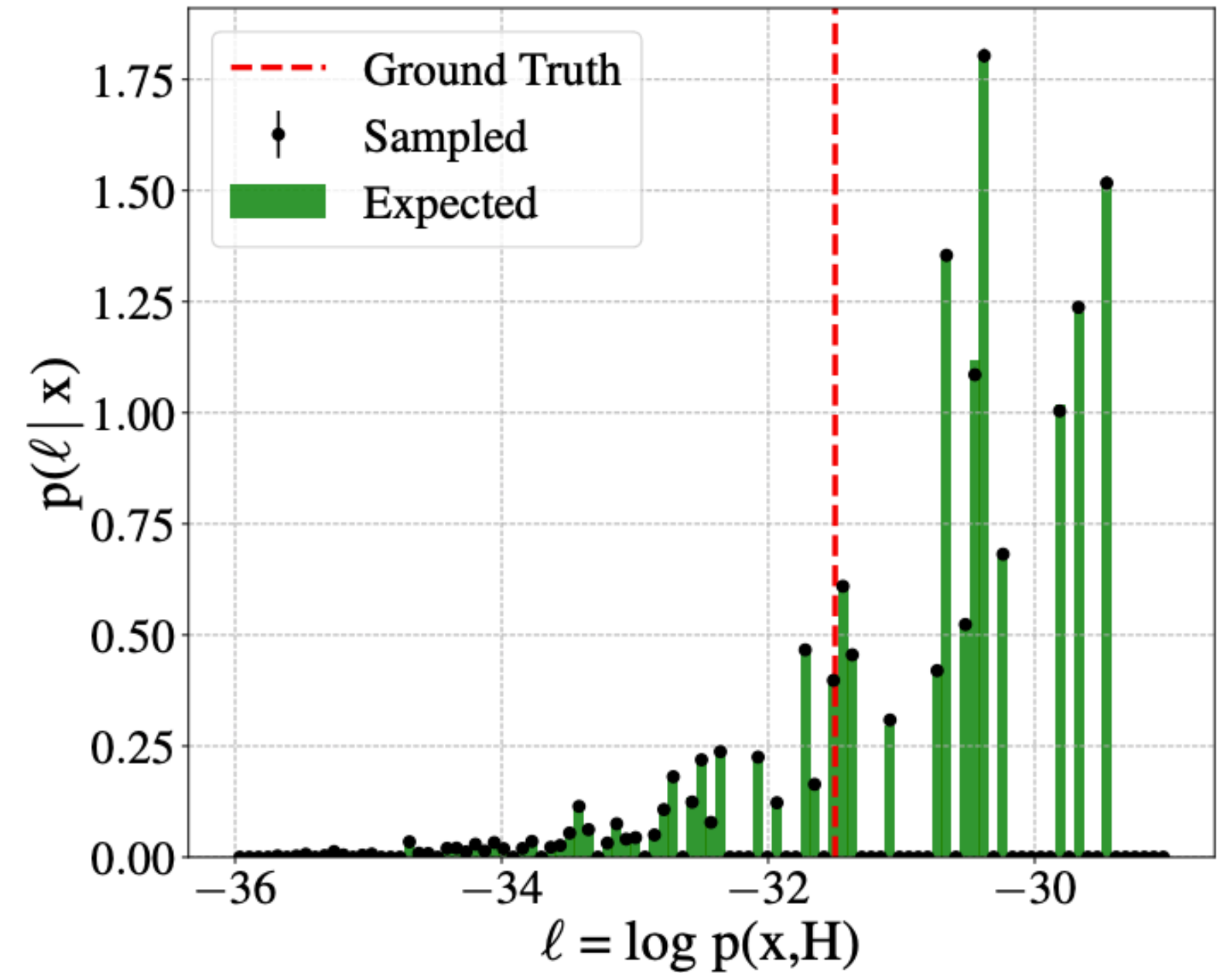
- We represent each set of elements as a node in the trellis.
- There are  $2^{|x|}$  nodes for a full trellis, but the number of trees grows super-exponentially.



# Posterior distribution - Sampling procedure

- The trellis encodes a distribution over all possible trees.
- Traversing the trellis top-bottom is similar to running the generative model.
- Trellis enables to sample histories weighted by their likelihood from the true posterior distribution conditioned on a set of leaves (jet constituents) without enumerating all possible clusterings.

$$p(z|x, \theta) = \frac{p(x|z, \theta) p(z|\theta)}{p(x|\theta)}$$



## Event Generation for events with large jet multiplicity

During simulations, when implementing the CKKW-L matching algorithm, parton final states need to be reweighted with the corresponding Sudakov form factors of each history  $p(x, z|\theta)$ .

Trellis could be extended to consider  $2 \rightarrow 3$  splittings (currently based on binary trees,  $1 \rightarrow 2$  splittings); could make feasible the implementation of CKKW-L to a higher jet multiplicity.