Consider a collection of weighted subsets of a ground set $N$. We present a tree-based greedy heuristic, Treedy, that for a given query subset $Q$ of $N$ and a tolerance $d$ approximates the weighted sum over all subsets of $Q$ within relative error $d$. It also enables approximate sampling of subset of $Q$ proportionally to the weights within total variation distance $d$. Experimental results show that approximations yield dramatic savings in running time compared to exact computation, and that Treedy typically outperforms a previously proposed sorting-based heuristic.

### Introduction

#### Problem Definition

**Input:** A downward closed collection $C$ of subsets of a ground set $N$. Weights $w(S) \geq 0$ for $S \in C$ and $w(S) = 0$ otherwise.

**Query:** Query set $Q \subseteq N$. Tolerance $d \geq 0$.

**Counting problem:** Approximate

$$W(Q) = \sum_{S \subseteq Q} w(S)$$

within relative error $d$.

**Sampling problem:** Draw a random subset $S \subseteq Q$ from a distribution within total variation distance $d$ from $Pr(S) = w(S)/W(Q)$.

#### Algorithms

"Collector algorithm" approach: Visit the subsets of $Q$ that are in $C$ (called relevant sets) and add up their weights until the sum is guaranteed to be within tolerance $d$.

**Algorithm: Exact**

A baseline method that visits all relevant sets. Computes the sum exactly.

**Algorithm: Ideal**

An idealized method that visits the minimum number of heaviest relevant sets to reach tolerance $d$. (Simulated in the experiments.)

**Algorithm: Sorted**

An improved version of the heuristic of Friedman and Koller for order-MCMC [1].

**Preprocessing:** Sorts the sets in $C$ by weight starting from the heaviest set.

**Per query:** Traverses the tree greedily w.r.t. $\psi$. Ignores irrelevant branches. Stops once the weight of remaining branches is small enough compared to accumulated weight.

**Application: Bayesian Network Learning**

Order-MCMC [1] is a method for learning the structure of a Bayesian network. It samples node orderings $v_1 v_2 \cdots v_n$ from the posterior distribution

$$Pr(v_1 v_2 \cdots v_n) = \prod_{i=1}^{n} W_i(\{v_1, \ldots, v_{i-1}\})$$

where $W_i(S) = \sum_{S \subseteq Q} w_i(S)$ is a sum over possible parent sets of $v_i$.

- optionally samples DAGs from orderings $\Rightarrow n$ subset sampling queries

**Algorithm: Treedy**

A novel heuristic based on tree traversal.

**Preprocessing:** Builds a "greedy tree" with $\phi$ and aggregate potentials $\psi$.

**Per query:** Traverses the tree greedily w.r.t. $\psi$. Ignores irrelevant branches. Stops once the weight of remaining branches is small enough compared to accumulated weight.

**From Counting to Sampling**

Sampling within total variation distance $d$ to the exact distribution is possible by first visiting relevant sets up to tolerance $d$ and then drawing samples from visited sets.

#### Experiments

We measured the time ($s$) per subset counting query as a function of approximation tolerance $d$. Parameter $k \in \{4, 5\}$ was used to restrict the size of the subsets in $C$.

**Artificial Instances**

Runtimes for different types of artificial weight functions ($n = 60, k = 5$):

**Bayesian Network Learning**

Runtime of order-MCMC for data from ALARM-network ($n = 37, k = 5$):

Runtime of order-MCMC for datasets from the UCI repository ($n \in \{34, 61\}, k \in \{4, 5\}$):