# Learning Hierarchical Riffle Independent Groupings from Rankings

**Jonathan Huang** JCH1@CS.CMU.EDU
**Carlos Guestrin** GUESTRIN@CS.CMU.EDU
Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA, 15213

## Abstract

Riffled independence is a generalized notion of probabilistic independence that has been shown to be naturally applicable to ranked data. In the riffled independence model, one assigns rankings to two disjoint sets of items independently, then in a second stage, interleaves (or riffles) the two rankings together to form a full ranking, as if by shuffling a deck of cards. Because of this interleaving stage, it is much more difficult to detect riffled independence than ordinary independence. In this paper, we provide the first automated method for discovering sets of items which are riffle independent from a training set of rankings. We show that our clustering-like algorithms can be used to discover meaningful latent coalitions from real preference ranking datasets and to learn the structure of hierarchically decomposable models based on riffled independence.

## 1. Introduction

Ranked data appears ubiquitously in various machine learning application domains. Rankings are useful, for example, in reasoning about preference lists in surveys, search results in information retrieval applications, and ballots in certain elections. The problem of building statistical models on rankings has thus been an important research topic in the learning community. As with many challenging learning problems, one must contend with an intractably large state space when dealing with rankings since there are $n!$ ways to rank $n$ objects. In building a statistical model over rankings, simple (yet flexible) models are therefore preferable because they are typically more computationally tractable, less prone to overfitting, and often more interpretable.

Appearing in *Proceedings of the 27$^{th}$ International Conference on Machine Learning*, Haifa, Israel, 2010. 

A popular and highly successful approach for achieving such simplicity for distributions involving large collections of interdependent variables has been to exploit conditional independence structures (e.g., with naive Bayes, tree, or Markov models). With ranking problems, however, independence-based relations are typically trickier to exploit due to the so-called *mutual exclusivity* constraints which constrain any two items to map to different ranks in any given ranking. In a recent paper, Huang and Guestrin (2009) proposed an alternative notion of independence, called *riffled independence*, which they have shown to be more natural for rankings. In addition to being a natural way to represent distributions in a factored form, the concept of riffled independence leads to a natural way of grouping items together; If we find two subsets of items, $A$ and $B$ to be riffle independent of each other, then it is often the case that items in $A$ (and items in $B$) are associated with each other in some way. Experimentally, Huang and Guestrin (2009) were able to demonstrate on a particular election dataset (of the American Psychological Association, (Diaconis, 1989)), that the political coalitions formed by candidates in the election were in fact approximately riffle independent of each other. However, it is not always obvious what kind of groupings riffled independence will lead to. Should fruits really be riffle independent of vegetables? Or are green foods riffle independent of red foods?

In this paper, we address the problem of automatically discovering optimal riffle independent groupings of items from training data. Key among our observations is the fact that while item ranks cannot be independent due to mutual exclusivity, relative ranks between sets of items are not subject to the same constraints. More than simply being a 'clustering' algorithm, however, our procedure can be thought of as a structure learning algorithm, like those from the graphical models literature, which find the optimal (riffled) independence decomposition of a distribution. Structure learning algorithms are useful for efficient probabilistic representations and inference, and in the spirit of graphical models, we explore a family of probabilistic models for

rankings based on hierarchies of groupings using riffled independence. We show that our hierarchical models are simple and interpretable, and present a method for performing model selection based on our algorithm. Our main contributions are as follows:

- We propose a method for finding the partitioning of an item set such that the subsets of the partition are as close to riffle independent as possible. In particular, we propose a novel objective for quantifying the degree to which two subsets are riffle independent to each other.
- We define a family of simple and interpretable distributions over rankings, called hierarchical riffle independent models, in which subsets of items are interleaved into larger subsets in a recursive stagewise fashion. We apply our partitioning algorithm to perform model selection from training data in polynomial time, without having to exhaustively search over the exponentially large space of hierarchical structures.

## 2. Riffled independence for rankings

In this paper we will be concerned with distributions over rankings. A *ranking* $\sigma = (\sigma(1), \ldots, \sigma(n))$ is a one-to-one association between $n$ items and ranks, where $\sigma(j) = i$ means that the $j^{th}$ item is assigned rank $i$ under $\sigma$. We will also refer to a ranking $\sigma$ by its inverse, $[\![\sigma^{-1}(1), \ldots, \sigma^{-1}(n)]\!]$ (called an *ordering* and denoted with double brackets instead of parentheses), where $\sigma^{-1}(i) = j$ also means that the $j^{th}$ item is assigned rank $i$ under $\sigma$. As a running example in this paper, we will consider ranking a small list of 6 items consisting of fruits and vegetables: (**C**) Corn, (**P**) Peas, (**L**) Lemons, (**O**) Oranges, (**F**) Figs, and (**G**) Grapes. The ordering $\sigma = [\![\mathbf{P}, \mathbf{F}, \mathbf{C}, \ldots]\!]$, for example, ranks Peas first, Figs second, Corn third and so on. A distribution $h(\sigma)$, defined over the set of rankings, $S_n$, can be viewed as a joint distribution over the $n$ variables $(\sigma(1), \ldots, \sigma(n))$ (where $\sigma(j) \in \{1, \ldots, n\}$), subject to *mutual exclusivity constraints* which stipulate that two objects cannot simultaneously map to the same rank, or alternatively, that two ranks cannot simultaneously be occupied by the same object ($h(\sigma(i) = \sigma(j)) = 0$ whenever $i \neq j$).

Since there are $n!$ rankings for $n$ items, representing arbitrary distributions $h$ is not viable for large $n$, and it is typical to restrict oneself to simplified classes of distributions. It might be tempting to think that probabilistic independence assertions would be a useful simplification for ranking problems (as naive Bayes models have been useful in other areas of machine learning). However, mutual exclusivity makes probabilistic independence a tricky notion for rankings, since each pair of items is constrained to map to different ranks. Another way to see this is to imagine constructing a graphical model representing $h$ in which each node corresponds to the rank of an item. For each pair of nodes, mutual exclusivity induces an edge between those two nodes, leading to a fully connected graph (Huang et al., 2009).

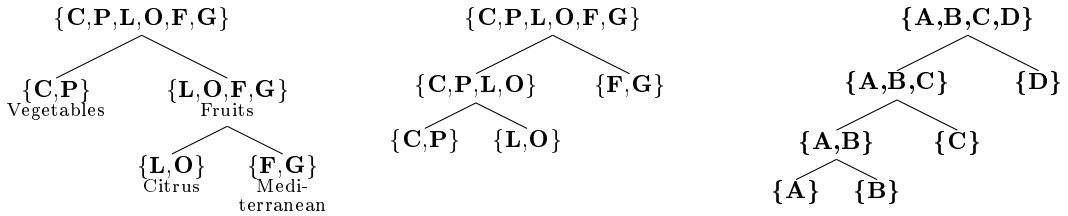
*Riffled independence*, introduced recently in (Huang & Guestrin, 2009), turns out to be a more natural notion of independence in the ranking setting. Consider our example of jointly ranking a set containing both vegetables (denoted by set $A$) and fruits (denoted by set $B$) in order of preference. The idea of riffled independence is to first draw two independent rankings (one for vegetables, one for fruits), then to interleave the two sets to form a full ranking for the entire collection. The word *riffle* comes from the popular shuffling technique called the *riffle shuffle* in which one cuts a deck of cards and interleaves the piles. For a formal definition, we establish the following notation about *relative rankings* and *interleavings*. Given a ranking $\sigma \in S_n$, and a subset $A \subset \{1, \ldots, n\}$, let $\phi_A(\sigma)$ denote the ranks of items in $A$ *relative* to the set $A$. For example, in the ranking $\sigma = [\![\mathbf{P}, \mathbf{L}, \mathbf{F}, \mathbf{G}, \mathbf{C}, \mathbf{O}]\!]$, the relative ranks of the vegetables is $\phi_A(\sigma) = [\![\mathbf{P}, \mathbf{C}]\!] = [\![Peas, Corn]\!]$. Thus, while corn is ranked fifth in $\sigma$, it is ranked second in $\phi_A(\sigma)$. Similarly, the relative ranks of the fruits is $\phi_B(\sigma) = [\![\mathbf{L}, \mathbf{F}, \mathbf{G}, \mathbf{O}]\!] = [\![Lemons, Figs, Grapes, Oranges]\!]$. Likewise, let $\tau_{A,B}(\sigma)$ denote the way in which the sets $A$ and $B$ are interleaved by $\sigma$. For example, using the same $\sigma$ as above, the interleaving of vegetables and fruits is $\tau_{A,B}(\sigma) = [\![Veg, Fruit, Fruit, Fruit, Veg, Fruit]\!]$. We will use $\Omega_{A,B}$ to denote the collection of all distinct ways of interleaving the sets $A$ and $B$. Note that if $|A| = k$, then $|\Omega_{A,B}| = \binom{n}{k}$.

**Definition 1** (Huang and Guestrin (2009))**.** Let $h$ be a distribution over $S_n$ and consider a subset $A \subset \{1, \ldots, n\}$ (with $|A| = k$)and its complement $B$. The sets $A$ and $B$ are said to be *riffle independent* if $h$ factors as:

$$h(\sigma) = m(\tau_{A,B}(\sigma)) \cdot f_A(\phi_A(\sigma)) \cdot g_B(\phi_B(\sigma)), \qquad (2.1)$$

where $m$, $f_A$, $g_B$ are distributions over $\Omega_{A,B}$, $S_k$, and $S_{n-k}$, respectively. We will refer to $m$ as the *interleaving distribution*, and to $f_A$ and $g_B$ as the *relative ranking distributions* of $A$ and $B$. We will also notate the relation as $A \perp_m B$.

Riffled independence can be seen as a generalization of full independence (when the interleaving distribution is deterministic). In addition to exploring theoretical properties, Huang and Guestrin showed




(a) Example of hierarchical riffled independence structure on $S_6$

(b) Another example, not equivalent to (a)

(c) Hierarchical decomposition into singleton subsets (each leaf set consists of a single item)

*Figure 1.* Examples of distinct hierarchical riffle independent structures

that the riffled independence relation in fact holds (approximately) in certain datasets. In particular, they showed on a particular election dataset, that the political coalitions in the candidate set were nearly riffle independent of each other. In other nonpolitical ranking datasets, however, it is not always obvious how one should partition the item set so that the riffle independent approximation is as close as possible to the true joint distribution. Is there any way to discover latent coalitions of items arising from riffled independence from ranked data? Before presenting our solution to this question, we discuss a more complicated but realistic scenario in which coalitions are further partitioned into smaller subgroups.

## 3. Hierarchical decompositions

For a fixed subset $A$ (and its complement $B$), there are $O\left(k! + (n-k)! + \binom{n}{k}\right)$ model parameters (specifying $f_A$, $g_B$, and $m$). Despite the fact that riffle independent models are much more compact compared to full models over rankings, it is still intractable to represent arbitrary riffle independent models for large $n$, and one must make further simplifications. For example, Huang and Guestrin discuss methods for representing useful parametric families of interleaving distributions, with just a handful of parameters.

We will explore the other natural model simplification which comes from the simple observation that the relative ranking distributions $f_A$ and $g_B$ are again distributions over rankings and so the sets $A$ and $B$ can further be decomposed into two riffle independent subsets. We call such models *hierarchical riffle independent decompositions*. Continuing with our running example, one can imagine that the fruits are further partitioned into two sets, a set consisting of citrus fruits ((**L**) Lemons and (**O**) Oranges) and a set consisting of mediterranean fruits ((**F**) Figs and (**G**) Grapes). To generate a full ranking, one first draws rankings of the citrus and mediterranean fruits independently ($[\![\mathbf{L}, \mathbf{O}]\!]$ and $[\![\mathbf{G}, \mathbf{F}]\!]$, for example). Secondly, the two sets are interleaved to form a ranking of all fruits ($[\![\mathbf{G}, \mathbf{L}, \mathbf{O}, \mathbf{F}]\!]$). Finally, a ranking of the vegetables is drawn ($[\![\mathbf{P}, \mathbf{C}]\!]$) and interleaved with the fruit rankings to form a full joint ranking: $[\![\mathbf{P}, \mathbf{G}, \mathbf{L}, \mathbf{O}, \mathbf{F}, \mathbf{C}]\!]$. Notationally, we can express the hierarchical decomposition as $\{\mathbf{P}, \mathbf{C}\} \perp_m (\{\mathbf{L}, \mathbf{O}\} \perp_m \{\mathbf{F}, \mathbf{G}\})$. We can also visualize hierarchies using trees (see Figure 1(a) for our example). The subsets of items which appear as leaves in the tree will be referred to as *leaf sets*.

A natural question to ask is: if we used a different hierarchy with the same leaf sets, would we capture the same distributions? For example, does a distribution which decomposes according to the tree in Figure 1(b) also decompose according to the tree in Figure 1(a)? The answer, in general, is no, due to the fact that distinct hierarchies impose different sets of independence assumptions, and as a result, different structures can be well or badly suited for modeling a given dataset. Consequently, it is important to use the "correct" structure if possible. We remark that while the two structures 1(a),1(b), capture distinct families of distributions, it is possible to identify a set of independence assumptions common to both structures.

## 4. Objective functions

Since different hierarchies impose different independence assumptions, we would like to find the structure that is best suited for modeling a given ranking dataset. The base problem that we now address is how to find the best structure if there is only one level of partitioning and two leaf sets, $A$, $B$. Alternatively, we want to find the topmost partitioning of the tree. In Section 5, we use this base case as part of a top-down approach for learning a full hierarchy.

**Problem statement.** Given a training set of rankings, $\sigma^{(1)}, \ldots, \sigma^{(m)} \sim h$, in which a subset of items, $A \subset \{1, \ldots, n\}$, is riffle independent of its complement, we would like to automatically determine the sets $A$ and $B \subset \{1, \ldots, n\}$. If $h$ does not *exactly* factor riffle

independently, then we would like to find the riffle independent approximation which is *closest* to $h$ in some sense. Formally, we would like to solve the problem:

$$\arg\min_{A} \min_{m,f,g} D_{KL}(\hat{h}(\sigma) \,||\, m(\tau_{A,B}(\sigma))f(\phi_A(\sigma))g(\phi_B(\sigma))), \tag{4.1}$$

where $\hat{h}$ is the empirical distribution of training examples and $D_{KL}$ is the Kullback-Leibler divergence measure. Equation 4.1 is a seemingly reasonable objective since it can also be interpreted as maximizing the likelihood of the training data. If $A$ and $B$ are truly riffle independent, then 4.1 can be shown via the Gibbs inequality to attain its minimum, zero, at (and only at) the subsets $A$ and $B$.

For small problems, one can actually solve Problem 4.1 using a single computer by evaluating the approximation quality of each subset $A$ and taking the minimum, which was the approach taken in (Huang & Guestrin, 2009). However, for larger problems, one runs into time and sample complexity problems since optimizing the globally defined objective function (4.1) requires relearning all model parameters ($m$, $f_A$, and $g_B$) for each of the exponentially many subsets of $\{1, \ldots, n\}$. In the remainder, we propose a more locally defined objective function, reminiscent of clustering, which we will use instead of Equation 4.1. As we show, our new objective will be more tractable to compute and have lower sample complexity for estimation.

**Proposed objective function.** The approach we take is to minimize a different measure that exploits the observation that *absolute ranks of items in $A$ are fully independent of relative ranks of items in $B$, and vice versa.* With our vegetables and fruits, for example, knowing that Figs is ranked first among all six items (the absolute rank of a fruit) should give no information about whether Corn is preferred to Peas (the relative rank of vegetables). More formally, given a subset $A = \{a_1, \ldots, a_\ell\}$, let $\sigma(A)$ denote the vector of (absolute) ranks assigned to items in $A$ by $\sigma$ (thus, $\sigma(A) = (\sigma(a_1), \sigma(a_2), \ldots, \sigma(a_\ell))$). We propose to minimize an alternative objective function:

$$\mathcal{F}(A) \equiv I(\sigma(A)\,;\,\phi_B(\sigma)) + I(\sigma(B)\,;\,\phi_A(\sigma)), \tag{4.2}$$

where $I$ denotes the mutual information (defined between two variables $X_1$ and $X_2$ by $I(X_1; X_2) \equiv D_{KL}(P(X_1, X_2)||P(X_1)P(X_2))$. We can show that $\mathcal{F}$ is guaranteed to detect riffled independence:

**Proposition 2.** *$\mathcal{F}(A) = 0$ is a necessary and sufficient criterion for a subset $A \subset \{1, \ldots, n\}$ to be riffle independent of its complement, $B$.*

As with Equation 4.1, optimizing $\mathcal{F}$ is still intractable for large $n$. However, it motivates a natural proxy,

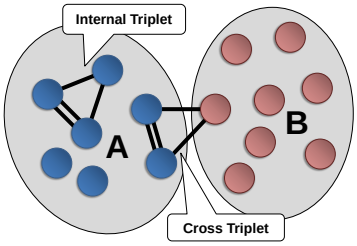



*Figure 2.* Graphical depiction of the problem of finding riffle independent subsets. Note the similarities to clustering. Double bars represent the direct nature of the triangles.

in which we replace the mutual informations defined over all $n$ variables by a sum of mutual informations defined over just three variables at a time. Given any triplet of distinct items, $(i, j, k)$, let $I_{i;j,k} \equiv I(\sigma(i)\,;\,\sigma(j) < \sigma(k))$. We define $\Omega^{cross}_{A,B}$ to be the set of triplets which "cross" from set $A$ to set $B$:

$$\Omega^{cross}_{A,B} \equiv \{(i; j, k) \,:\, i \in A, j, k \in B\}.$$

$\Omega^{cross}_{B,A}$ is similarly defined. We also define $\Omega^{int}_{A}$ to be the set of triplets that are internal to $A$

$$\Omega^{int}_{A} \equiv \{(i; j, k) \,:\, i, j, k \in A\},$$

and again, $\Omega^{int}_{B}$ is similarly defined. Our proxy objective function can be written as the sum of the mutual information evaluated over all of the crossing edges:

$$\tilde{\mathcal{F}}(A) \equiv \sum_{(i,j,k)\in\Omega^{cross}_{A,B}} I_{i;j,k} + \sum_{(i,j,k)\in\Omega^{cross}_{B,A}} I_{i;j,k}. \tag{4.3}$$

$\tilde{\mathcal{F}}$ can be viewed as a low order version of $\mathcal{F}$, involving mutual information computations over triplets of variables at a time instead of $n$-tuples. The mutual information $I_{a;b,b'}$, for example, reflects how much the rank of a vegetable tells us about how two fruits compare. If $A$ and $B$ are riffle independent, then we know that $I_{a;b,b'} = 0$ and $I_{b;a,a'} = 0$ for any items $a, a' \in A$ and $b, b' \in B$. The objective $\tilde{\mathcal{F}}$ is somewhat reminiscent of typical graphcut and clustering objectives. Instead of partitioning a set of nodes based on sums of pairwise similarities, we partition based on sums of tripletwise affinities. We show a graphical depiction of the problem in Figure 2, where cross triplets (in $\Omega^{cross}_{A,B}$, $\Omega^{cross}_{B,A}$) have low weight and internal triplets (in $\Omega^{int}_{A}$, $\Omega^{int}_{B}$) have high weight. The objective is to find a partition such that the sum over cross triplets is low. In fact, the problem of optimizing $\tilde{\mathcal{F}}$ can be seen as an instance of the weighted, directed hypergraph cut problem (Gallo et al., 1993). Note that the word *directed* is significant for us, because, unlike typical clustering problems, our triplets are not symmetric (for example, $I_{i;jk} \neq I_{j;ik}$), resulting in a nonstandard and poorly understood optimization problem.

**Third-order detectability assumptions.** When does $\tilde{\mathcal{F}}$ detect riffled independence? It is not difficult

to see, for example, that $\tilde{\mathcal{F}} = 0$ is a necessary condition for riffled independence, since $A \perp_m B$ implies $I_{a;b,b'} = 0$. We have:

**Proposition 3.** *If $A$ and $B$ are riffle independent sets, then $\tilde{\mathcal{F}}(A) = 0$.*

However, the converse of Prop. 3 is not true in full generality without accounting for dependencies that involve larger subsets of variables. Just as the pairwise independence assumptions (that are commonly used for randomized algorithms (Motwani & Raghavan, 1996))[1] do not imply full independence between two sets of variables, there exist distributions which "look" riffle independent from tripletwise marginals but do not factor upon examining higher-order terms. Nonetheless, in most practical scenarios, we expect $\tilde{\mathcal{F}} = 0$ to imply riffled independence.

**Estimating $\tilde{\mathcal{F}}$ from samples.** We have so far argued that $\tilde{\mathcal{F}}$ is a reasonable function for finding riffle independent subsets. However, since we only have access to samples rather than the true distribution $h$ itself, it will only be possible to compute an approximation to the objective $\tilde{\mathcal{F}}$. In particular, for every triplet of items, $(i, j, k)$, we must compute an estimate of the mutual information $I_{i;j,k}$ from i.i.d. samples drawn from $h$, and the main question is: how many samples will we need in order for the approximate version of $\tilde{\mathcal{F}}$ to remain a reasonable objective function?

In the following, we denote the estimated value of $I_{i;j,k}$ by $\hat{I}_{i;j,k}$. For each triplet, we use a regularized procedure due to (Höffgen, 1993) to estimate mutual information. We adapt his sample complexity bound to our problem below.

**Lemma 4.** *For any fixed triplet $(i, j, k)$, the mutual information $I_{i;j,k}$ can be estimated to within an accuracy of $\Delta$ with probability at least $1 - \gamma$ using $S(\Delta, \gamma) \equiv O\left(\frac{n^2}{\Delta^2} \log^2 \frac{n}{\Delta} \log \frac{n^4}{\gamma}\right)$ i.i.d. samples and the same amount of time.*

The approximate objective function is therefore:

$$\hat{\mathcal{F}}(A) \equiv \sum_{(i,j,k) \in \Omega^{cross}_{A,B}} \hat{I}_{i;j,k} + \sum_{(i,j,k) \in \Omega^{cross}_{B,A}} \hat{I}_{i;j,k}.$$

What we want to now show is that, if there exists a unique way to partition $\{1, \ldots, n\}$ into riffle independent sets, then given enough training examples, our approximation $\hat{\mathcal{F}}$ uniquely singles out the correct partition as its minimum with high probability. A class of riffle independent distributions for which the uniqueness requirement is satisfied consists of the distributions for which $A$ and $B$ are *strongly connected* according to the following definition.

**Definition 5.** A subset $A \subset \{1, \ldots, n\}$ is called *$\epsilon$-third-order strongly connected* if, for every triplet $i, j, k \in A$ with $i, j, k$ distinct, we have $I_{i;j,k} > \epsilon$.

If a set $A$ is riffle independent of $B$ and both sets are third order strongly connected, then we can ensure that riffled independence is detectable from third-order terms and that the partition is unique. We have the following probabilistic guarantee.

**Theorem 6.** *Let $A$ and $B$ be $\epsilon$-third order strongly connected riffle independent sets, and suppose $|A| = k$. Given $S(\Delta, \epsilon) \equiv O\left(\frac{n^4}{\epsilon^2} \log^2 \frac{n^2}{\epsilon} \log \frac{n^4}{\gamma}\right)$ i.i.d. samples, the minimum of $\hat{\mathcal{F}}$ is achieved at exactly the subsets $A$ and $B$ with probability at least $1 - \gamma$.*

If $k$ is small compared to $n$, then one can actually derive a better bound: $S(\Delta, \epsilon) \equiv O\left(\frac{n^2}{\epsilon^2} \log^2 \frac{n^2}{\epsilon} \log \frac{n^4}{\gamma}\right)$.

**Finding balanced partitions.** We conclude this section with a practical extension of the basic objective function $\tilde{\mathcal{F}}$. In practice, like the minimum cut objective for graphs, the tripletwise objective of Equation 4.3 has a tendency to "prefer" small partitions (either $|A|$ or $|B|$ very small) to more balanced partitions ($|A|, |B| \approx n/2$) due to the fact that unbalanced partitions have fewer triplets that cross between $A$ and $B$. The simplest way to avoid this bias is to optimize the objective function over subsets of a fixed size $k$. As we discuss in the next section, optimizing with a fixed $k$ can be useful for building "thin" hierarchical riffle independent models. Alternatively, one can use a modified objective function that encourages more balanced partitions. For example, we use a variation of our objective function (inspired by the normalized cut criterion (Shi & Malik, 2000)) which has proven to be useful for detecting riffled independence when the size $k$ is unknown.

## 5. Structure discovery algorithms.

Having now designed a function that is tractable to estimate from both perspectives of computational and sample complexity, we turn to the problem of learning the hierarchical riffle independence structure from training examples. In this paper, we take a simple top-down approach in which the item sets are recursively partitioned by optimizing $\hat{\mathcal{F}}$ until some stopping criterion is met. The question is: can $\hat{\mathcal{F}}$ be optimized efficiently? We begin by discussing a restricted class of "thin" models for which we can tractably optimize $\hat{\mathcal{F}}$.

[1] A pairwise independent family of random variables is one in which any two members are marginally independent. Subsets with larger than two members may not necessarily factor independently, however.

**Learning "thin" chain models.** By a $k$-thin chain model, we refer to a hierarchical structure in which the size of the smaller set at each split in the hierarchy is fixed to be a constant $k \sim O(1)$ and can therefore be expressed as:

$$(A_1 \perp_m (A_2 \perp_m (A_3 \perp_m \dots))),\ |A_i| = k, \text{for all } i.$$

We view thin chains as being somewhat analogous to thin junction tree models (Chechetka & Guestrin, 2007), in which cliques are never allowed to have more than $k$ variables. To draw rankings from a thin chain model, one sequentially inserts items independently, one group of size $k$ at a time, into the full ranking. The structure learning problem is therefore to discover how the items are partitioned into groups, which group is inserted first, which group is inserted second, and so on.

When the size $k$ is known, the optimal partitioning of an item set can be found by exhaustively evaluating $\hat{\mathcal{F}}$ over all $k$-subsets. Finding the global optimum of $\hat{\mathcal{F}}$ is therefore guaranteed at each stage of the recursion. The time complexity for finding the optimal partition from $m$ samples is $O(kn^{k+2}+mn^3)$, accounting for precomputing the necessary mutual informations as well as optimization time, and is tractable if $k$ is small.

**Handling arbitrary partitions using anchors.** When $k$ is large, or even unknown, $\hat{\mathcal{F}}$ cannot be optimized using exhaustive methods. Instead, we propose a simple algorithm for finding $A$ and $B$ based on the following observation. If an oracle could identify any two elements of the set $A$, say, $a_1, a_2$, in advance, then the quantity $I_{x;a_1,a_2} = I(x; a_1 < a_2)$ indicates whether the item $x$ belongs to $A$ or $B$ since $I_{x;a_1,a_2}$ is nonzero in the first case, and zero in the second case.

For finite training sets, when $I$ is only known approximately, one can sort the set $\{I_{x;a_1,a_2}\,;\, x \neq a_1, a_2\}$ and if $k$ is known, take the $k$ items closest to zero to be the set $B$. Since we compare all items against $a_1, a_2$, we refer to these two fixed items as "anchors". Of course $a_1, a_2$ are not known in advance, but the above method can be repeated for every pair of items as anchors to produce a collection of $O(n^2)$ candidate partitions. Each partition can then be scored using the approximate objective $\hat{\mathcal{F}}$, and a final optimal partition can be selected as the minimum over the candidates. See Algorithm 1. In cases when $k$ is not known a priori, we evaluate partitions for all possible settings of $k$ using $\hat{\mathcal{F}}$.

Since the anchors method does not require searching over subsets, it can be significantly faster (with $O(n^3 m)$ time complexity) than an exhaustive optimization of $\hat{\mathcal{F}}$. Moreover, by assuming $\epsilon$-third order strong connectivity as in the previous section, one can use similar arguments to derive sample complexity bounds. We remark that there are practical differences that can at times make the anchors method somewhat less robust than an exhaustive search. Conceptually, anchoring works well when there exists two elements that are strongly connected with all of the other elements in its set, whereas an exhaustive search can work well in weaker conditions such as when items are strongly connected through longer paths. We show in our experiments that the anchors method can nonetheless be quite effective for learning hierarchies.

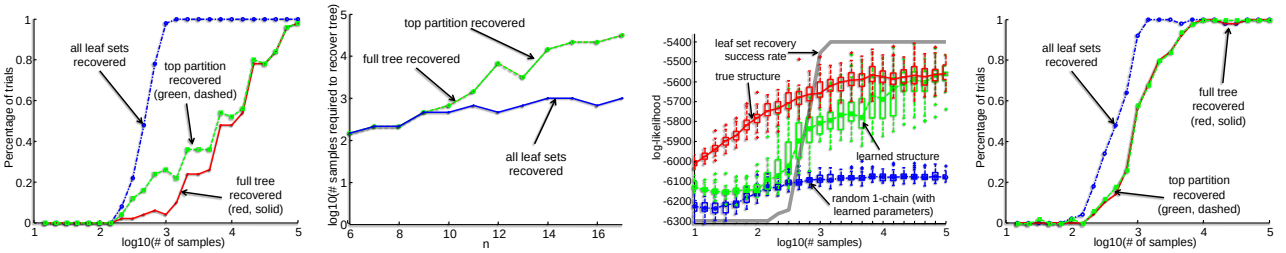
**Algorithm 1** The Anchors method

**Input:** training set $\{\sigma^{(1)}, \dots, \sigma^{(m)}\}$, $k \equiv |A|$.
Estimate and cache $\hat{I}_{i;j,k}$ for all $i, j, k$;
**for all** $a_1, a_2 \in \{1, \dots, n\},\ a_1 \neq a_2$ **do**
  $\hat{I}^k \leftarrow k^{th}$ smallest item in $\{\hat{I}_{x;a_1,a_2}; x \neq a_1, a_2\}$;
  $A_{a_1,a_2} \leftarrow \{x : \hat{I}_{x;a_1,a_2} \leq \hat{I}^k\}$ ;
**end for**
$A_{best} \leftarrow \arg\min_{a_1,a_2} \hat{\mathcal{F}}(A_{a_1,a_2})$;
**output** $A_{best}$;

## 6. Experiments

**Synthetic data.** We first applied our methods to synthetic data to show that, given enough samples, our algorithms *do* effectively recover the optimal hierarchical structures. For various settings of $n$, we simulated data drawn jointly from a $k$-thin chain model (for $k = 4$) with a random parameter setting for each structure and applied our exact method for learning thin chains to each sampled dataset. First, we investigated the effect of varying sample size on the proportion of trials (out of fifty) for which our algorithms were able to (a) recover the full underlying tree structure *exactly*, (b) recover the topmost partition correctly, or (c) recover all leaf sets correctly (but possibly out of order). Figure 3(a) shows the result for an itemset of size $n = 16$. Figure 3(b), shows, as a function of $n$, the number of samples that were required in the same experiments to (a) *exactly* recover the full underlying structure or (b) recover the correct leaf sets, for at least 90% of the trials. What we can observe from the plots is that, given enough samples, reliable structure recovery *is* indeed possible. It is also interesting to note that recovery of the correct leaf sets can be done with much fewer samples than are required for recovering the full hierarchical structure of the model.

After learning a structure for each dataset, we learned model parameters and evaluated the log-likelihood of each model on 200 test examples drawn from the true distributions. In Figure 3(c), we compare log-




(a) Success rate for structure recovery vs. sample size ($n = 16, k = 4$)

(b) Number of samples required for structure recovery vs. number of items $n$

(c) Test set log-likelihood comparison

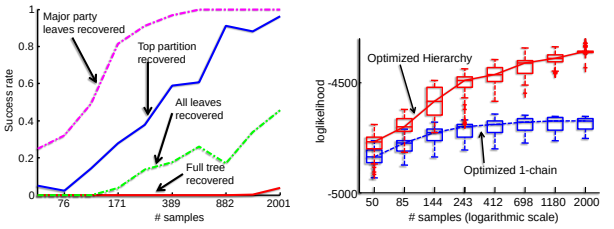
(d) Anchors algorithm success rate ($n = 16$, unknown $k$)

*Figure 3.* Simulated data experiments

likelihood performance when (a) the true structure is given (but not parameters), (b) a $k$-thin chain is learned with known $k$, and (c) when we use a random generated 1-chain structure. As expected, knowing the true structure results in the best performance, and the 1-chain is overconstrained. However, our structure learning algorithm is eventually able to catch up to the performance of the true structure given enough samples. It is also interesting to note that the jump in performance at the halfway point in the plot coincides with the jump in the success rate of discovering all leaf sets correctly — we conjecture that performance is sometimes less sensitive to the actual hierarchy used, as long as the leaf sets have been correctly discovered.

To test the Anchors algorithm, we ran the same simulation using Algorithm 1 on data drawn from hierarchical models with no fixed $k$. We generated roughly balanced structures, meaning that item sets were recursively partitioned into (almost) equally sized subsets at each level of the hierarchy. From Figure 3(d), we see that the Anchors algorithm can also discover the true structure given enough samples. Interestingly, the difference in sample complexity for discovering leaf sets versus discovering the full tree is not nearly as pronounced as in Figure 3(a). We believe that this is due to the fact that the balanced trees have less depth than the thin chains, leading to fewer opportunities for our greedy top-down approach to commit errors.

**Election data.** We now demonstrate our methods with real datasets. A number of electoral systems around the world require voters to provide a ranking of a set of candidates in order of preference. Diaconis (1989), for example, analyzed a 1980 presidential election of the American Psychological Association (APA) consisting of 5738 rankings of five candidates (W. Bevan, I. Iscoe, C. Kiesler, M. Siegle, and L. Wright). We applied our methods to learning a hierarchy from the APA data. Since there are only five candidates, we tried both the exhaustive optimization algorithm as well as the anchors algorithm. Both




(a) Stable features from recovered hierarchies vs. sample size

(b) Test log-likelihood vs. sample size

*Figure 4.* Irish election data experiments

methods recovered the same structure in roughly the same time (shown in Figure 5), which is also the structure that (Huang & Guestrin, 2009) obtained by performing a brute-force search over partitions using the KL-divergence objective (Equation 4.1). The leaf sets of the learned hierarchy in fact reflect three coalitions in the APA made up of research, clinical and community psychologists respectively.

Next, we applied our algorithms to a larger Irish House of Parliament (Dáil Éireann) election dataset from the Meath constituency in Ireland. See Gormley and Murphy (2006) for more election details (including candidate names) as well as an alternative analysis. There were 14 candidates in the 2002 election belonging to the two major rival political parties, Fianna Fáil and Fine Gael, as well as a number of smaller parties. We used a subset of 2500 fully ranked ballots from the election. As with the APA data, both the exhaustive optimization of $\hat{\mathcal{F}}$ and the anchors algorithm returned the same tree, with running times of 69.7 seconds and 2.1 seconds respectively (not including the 3.1 seconds required for precomputing mutual informations). The resulting tree, with candidates enumerated alphabetically from 1 through 14, is shown (only up to depth 4), in Figure 6. As expected, the candidates belonging to the two major parties, Fianna Fáil and Fine Gael, are neatly partitioned into their own leaf sets. The topmost leaf is the Sinn Fein candidate, indicating

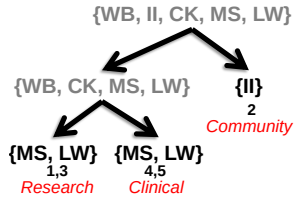



*Figure 5.* Hierarchy learned from APA election data

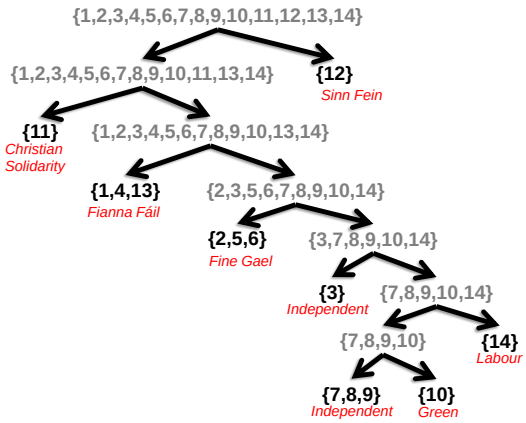



*Figure 6.* Hierarchy learned from Irish election data. See (Gormley & Murphy, 2006) for a list of candidates. The candidates are ordered alphabetically in this tree.

that voters tended to insert him into the ranking independently of all of the other 13 candidates.

To understand the behavior of our algorithm with smaller sample sizes, we looked for features of the tree from Figure 6 which remained stable even when learning with smaller sample sizes. In Figure 4(a), we subsample from the original training set 100 times at different sample sizes and plot the proportion of learned hierarchies which, (a) recover the Sinn Fein candidate as the topmost leaf, (b) partition the two major parties into leaf sets, and (c) agree with the original tree on all leaf sets. Note that even with few training examples, candidates belonging to the major parties are correctly grouped together indicating strong party influence in voting behavior.

Finally, we compared the results between learning a general hierarchy (without fixed $k$) and learning a 1-thin chain model on the Irish data. Figure 4(b) shows the log-likelihoods achieved by both models on a held-out test set as the training set size increases. For each training set size, we subsampled the Irish dataset 100 times to produce confidence intervals. Again, even with small sample sizes, the hierarchy outperforms the 1-chain and continually improves with more and more training data. One might think that the hierarchical models, which use more parameters are prone to overfitting, but in practice, the models learned by our algorithm devote most of the extra parameters towards modeling the correlations among the two major parties. As our results suggest, such intraparty ranking correlations are crucial for achieving good modeling performance.

## 7. Conclusion

We have investigated an intuitive ranking model based on hierarchical riffled independence decompositions and have shown that the structure of such hierarchies can be efficiently learned from data using our proposed algorithm which can automatically find riffle independent partitions within item sets.

Like Bayesian networks, cliques of variables for hierarchical riffled independence models can often form intuitive groupings, such as in the election datasets that we analyzed in this paper. However, for problems in which the item groupings are nonobvious, structure learning is of fundamental importance, and we believe that our contributions in this paper open the door to further research into the problem of decomposing huge distributions into tractable "pieces" much like Bayesian networks have done for other distributions.

### Acknowledgements

This work is supported in part by the ONR under MURI N000140710747, and the Young Investigator Program grant N00014-08-1-0752. We thank K. El-Arini for feedback, and B. Murphy and C. Gormley for datasets. Discussions with M. Meila provided valuable ideas upon which this work is based.

## References

Chechetka, A. and Guestrin, C. Efficient principled learning of thin junction trees. In *Advances in Neural Information Processing Systems 21*, 2007.

Diaconis, P. A generalization of spectral analysis with application to ranked data. *The Annals of Statistics*, 17 (3):949–979, 1989.

Gallo, G., Longo, G., Pallottino, S., and Nguyen, S. Directed hypergraphs and applications. *Discrete Applied Math.*, 42(2-3), 1993.

Gormley, C. and Murphy, B. A latent space model for rank data. In *23rd International Conference on Machine Learning*, 2006.

Höffgen, K. U. Learning and robust learning of product distributions. In *Sixth Annual Conference on Learning Theory*, 1993.

Huang, J. and Guestrin, C. Riffled independence for ranked data. In *Advances in Neural Information Processing Systems 23*, 2009.

Huang, J., Guestrin, C., Jiang, X., and Guibas, L. Exploiting probabilistic independence for permutations. In *Artificial Intelligence and Statistics, JMLR W&CP 5*, 2009.

Motwani, R. and Raghavan, P. Randomized algorithms. *ACM Computational Survey*, 28(1), 1996.

Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 22(8), 2000.