

Submitted to *Operations Research*

# Active Learning for Non-Parametric Choice Models

Fransisca Susan, Negin Golrezaei

MIT Sloan School of Management, Operations Management, fsusan@mit.edu, golrezae@mit.edu

Ehsan Emamjomeh-Zadeh

Meta Platforms, Inc., ehsan7069@gmail.com

David Kempe

University of Southern California, Los Angeles, david.m.kempe@gmail.com

Authors are encouraged to submit new papers to INFORMS journals by means of a style file template, which includes the journal title. However, use of a template does not certify that the paper has been accepted for publication in the named journal. INFORMS journal templates are for the exclusive purpose of submitting to an INFORMS journal and are not intended to be a true representation of the article's final published form. Use of this template to distribute papers in print or online or to submit papers to another non-INFORM publication is prohibited.

## Abstract.

We study the problem of actively learning a non-parametric choice model based on consumers' decisions. We present a negative result showing that such choice models may not be identifiable. To overcome the identifiability problem, we introduce a directed acyclic graph (DAG) representation of the choice model. This representation provably encodes all the information about the choice model which can be inferred from the available data, in the sense that it permits computing all choice probabilities.

We establish that given exact choice probabilities for a collection of item sets, one can reconstruct the DAG. However, attempting to extend this methodology to estimate the DAG from noisy choice frequency data obtained during an active learning process leads to inaccuracies. To address this challenge, we present an inclusion-exclusion approach that effectively manages error propagation across DAG levels, leading to a more accurate estimate of the DAG. Utilizing this technique, our algorithm estimates the DAG representation of an underlying non-parametric choice model. The algorithm operates efficiently (in polynomial time) when the set of frequent rankings is drawn uniformly at random. It learns the distribution over the most popular items among frequent preference types by actively and repeatedly offering assortments of items and observing the chosen item. We demonstrate that our algorithm more effectively recovers a set of frequent preferences on both synthetic and publicly available datasets on consumers' preferences, compared to corresponding non-active learning estimation algorithms. These findings underscore the value of our algorithm and the broader applicability of active-learning approaches in modeling consumer behavior.

**Key words:** Non-parametric choice models, Active learning, DAG representation

## 1. Introduction

Choice models are used by firms to capture consumers' preferences, and predict their decisions. They can help with important operational decisions, including demand forecasting (e.g., McFadden et al. (1977), McGill and Van Ryzin (1999)), inventory planning (e.g., Ryzin and Mahajan (1999), Gaur and Honhon (2006), Aouad et al. (2019)), assortment optimization (e.g., Talluri and Van Ryzin (2004), Rusmevichientong et al. (2010), Davis et al. (2014), Golrezaei et al. (2014)), and product ranking optimization (e.g., Derakhshan et al. (2020), Niazadeh et al. (2021), Golrezaei et al. (2021)). Among choice models, parametric choice models — and in particular random utility maximization models — have received the most attention by far. While parametric choice models provide concise representations of consumer preferences and decisions, they impose certain structures on consumer preferences, which may not be valid (Jagabathula and Rusmevichientong (2019)). If the structures are invalid, imposing them can lead to model misspecification and inaccurate decision making. These drawbacks have motivated the introduction and study of non-parametric choice models (Farias et al. 2013, Jagabathula and Rusmevichientong 2017, Paul et al. 2018)). Non-parametric choice models, however, are challenging to estimate using only offline collected transaction data since such data may not have enough variation in the offered sets of items.

In many settings, particularly on online platforms, one can go beyond using offline data. On these platforms, one can influence the data collection process, obtaining data sets more suitable for estimating non-parametric choice models. The process of influencing data collection, which is known as *active learning*, has been widely used in various contexts; see Settles (2009) and Aggarwal et al. (2014) for surveys, and Section 2 for a more detailed discussion of related work. In this process, the platform can dynamically change the set of items offered to arriving consumers and observe their choice. However, it is still unclear whether active learning can fully overcome the challenge of estimating non-parametric choice models. This motivates our main research question: *How can we estimate a non-parametric choice model by relying on active learning? Can we estimate non-parametric choice models with a polynomial-size dataset, which is obtained through an efficient active learning process?*

To answer these questions, we consider an online platform with  $n$  items. The platform faces distinct types of consumers: each consumer type is characterized by a preference ranking over the  $n$  items and the fraction of the population having this type. Consumers choose according to their types/rankings, and their types are not observable by the platform. The model is presented more formally and in detail in Section 3.

To learn the probability distribution over the rankings, the platform engages in an active learning process. In every round, a consumer (whose unobservable type is drawn based on the unknown population distribution) arrives. Upon the arrival of the consumer, based on past observations, the platform decides on the set (assortment) of items to offer. The consumer then chooses the item she ranks highest among the items offered. The platform observes the choice made by the consumer; then, the process repeats with another consumer.

## 1.1. Our Contributions

**Indistinguishability.** One of the main challenges in learning non-parametric choice models is the identifiability problem: the data may not uniquely identify the choice model. This problem clearly exists when the choice model is estimated from *offline* transaction data that contains the choices made by consumers only for some pre-specified offered sets; for example, if all consumers were offered the same set  $S$ , the algorithm can only learn the distribution of the most preferred item in  $S$ . With active learning, however, the algorithm can introduce heterogeneity in the offered sets. This raises the following question: Is it possible to learn *every* non-parametric choice model using active learning? In Section 4, we show that even with active learning, regardless of how many consumers the algorithm interacts with, the choice model may not be uniquely identifiable. In particular, we determine simple conditions under which two different non-parametric choice models are information-theoretically *indistinguishable* from each other, no matter how many and what assortments are offered to arriving consumers.

**Directed Acyclic Graph representations of non-parametric choice models.** In light of our indistinguishability result, in Section 5, we provide a novel representation of non-parametric choice models. This representation, which we refer to as Directed Acyclic Graph (DAG) representation, can always be uniquely identified, assuming enough samples with suitably chosen choice sets. The DAG contains a node for each set  $A$  of items such that at least one consumer type ranks all items in  $A$  (in any order) ahead of all items not in  $A$ ; thus, the nodes correspond to all possible prefixes of consumer rankings. The node  $A$  is labeled with the total probability of all types whose rankings have  $A$  as a prefix. All edges are of the form  $(A, A \cup \{z\})$  for an item  $z \notin A$ , and exist whenever there exists a consumer type ranking all of  $A$  in the first  $|A|$  positions (in any order), followed by  $z$  as the next item. For any  $j$ , the distribution over sets  $A$  of cardinality  $|A| = j$  captures the distribution over top- $j$  most preferred item sets; hence, it can be used in various decision-making processes, including inventory planning and product design decisions. More detail on such uses can be found in Section 8.

**Computing the choice probabilities using the DAG.** One of the key reasons for using the DAG representation is that in addition to giving an intuitive visual representation with good downstream processing properties, it still fully encodes all available information about the choice model. Specifically, as we will demonstrate in Section 6.1, when provided with a DAG representing a non-parametric choice model, one can — in polynomial time — calculate the probability of selecting an item  $z$  from a given set  $S$  of items. Since the only source of information for constructing the DAG representation is these choice frequencies, the DAG representation encapsulates as much information as can be gleaned from the available data.

**Computing the DAG using exact choice probabilities.** Given the desirability of having access to a DAG representation, we show — in Section 6.2 — that given the exact choice probabilities, an efficient algorithm can construct the DAG. The algorithm constructs the DAG iteratively, by increasing size of the prefix  $A$ .

When computing the relevant probabilities for larger prefixes  $A$ , it uses the probabilities of smaller prefixes  $A'$  whose probabilities were computed in earlier iterations.

**Active learning of the DAG representation.** Section 6.2 shows how to construct the DAG when exact choice probabilities are available. In practice, however, exact probabilities are inaccessible — instead, choice *frequencies* will be inferred from data. As we show, even small misestimates of choice frequencies have the potential to accumulate and lead to significant misestimates of the model. Concretely, this issue arises because probability estimates for larger prefixes use the estimates for smaller prefixes, which in turn use estimates for even smaller prefixes, etc., resulting in an inclusion-exclusion calculation. This type of dependency can lead to exponential amplification of estimation errors.

Our primary technical contribution lies in our method of selecting a significantly smaller and less “obvious” collection of sets for inclusion-exclusion operations. We achieve this by employing a set cover approximation algorithm on carefully chosen prefixes of  $A$ . When the set covers remain sufficiently small (of logarithmic size), the algorithm can identify the correct DAG with high accuracy using only a polynomial number of active queries, while still leading to only limited error propagation across different levels. Among others, the set covers are logarithmically small in the number  $n$  of items with high probability when the frequent items are uniformly random. When the set covers become larger, our algorithm exhibits adaptability. It can respond by using additional queries or by diagnosing situations where outputs obtained with few queries may not be reliable. This distinguishing feature sets our approach apart from parametric models, under which inaccuracies may arise if modeling assumptions are violated. The algorithm and its detailed analysis are comprehensively discussed in Section 7.

**Value of active learning via a case study.** To evaluate the accuracy and simplicity of our algorithm, we test it empirically on synthetically generated choice models as well as ranked preference parameters inferred from a Sushi preference data set (Kamishima (2003), Kamishima et al. (2005)); the experiments are presented in Section 9. We show that empirically, the number of data points needed to estimate the DAG representation of a choice model accurately is smaller than the theoretical bound, which shows the robustness of our algorithm. We also observe that our algorithm significantly outperforms a non-active choice model estimation algorithm (Farias et al. (2013)), while also using a smaller number of queries. This shows the value of our algorithm and the active learning approach in estimation.

## 2. Related Work

**Parametric choice models.** There are two general approaches to modeling consumer preferences: parametric and non-parametric models. Parametric choice models specify some functional form that connects related attributes and price to utility values and choice probabilities. One example is a choice model that assumes independent demand for each product; such a model does not capture substitution effects between

similar products in the offered assortment (Mahajan and van Ryzin 1999). Another example is the Multinomial Logit model (MNL), a parametric choice model that is commonly used in marketing, economics (see Ben-Akiva et al. (1985) and Mahajan and van Ryzin (1999)), and revenue management (McFadden 1973, Ben-Akiva et al. 1985). MNL has the IIA (independence of irrelevant alternatives) property: the odds of choosing one item over another do not depend on whether or not a third item is present in the assortment. This property is frequently unrealistic, especially when products exhibit complementarities. Other examples of parametric choice models, such as the generalized nested logit model (Wen and Koppelman 2001) and mixed logit model (Ben-Akiva et al. 1985), overcome the IIA restriction.

Another class of models, which also overcome the IIA restriction, is based on Markov Chains<sup>1</sup> (Blanchet et al. 2016, Feldman and Topaloglu 2017, Şimşek and Topaloglu 2018, Ma 2023). Under such models, the distribution over rankings  $\pi$  is defined as follows. The states of the Markov Chain are the items (plus a no-purchase option). The top choice is the starting node, drawn from a distribution  $(\lambda_j)$  over states  $j$ . Subsequently, the next state is always a node  $i$  chosen with conditional probability  $\rho_{j,i}$ . The ranking  $\pi$  is obtained as the order in which nodes were visited for the first time (since they will typically be visited multiple times). This model could in principle be encoded in a DAG — however, because all types have positive probability of occurring, the corresponding DAG would have  $2^n$  nodes.

Historically, there is a vast literature on optimizing the assortment and price based on some parametric choice models. This includes the seminal work by Talluri and Van Ryzin (2004) that infers a revenue-ordered set (a set containing a certain number of items with the highest revenue) as an optimal assortment for the deterministic MNL model, the work of Rusmevichientong et al. (2010) determining the optimal assortment for the MNL model with a capacity constraint, the work of Rusmevichientong and Topaloglu (2012) who develop an optimal assortment for a robust set of likely parameters of choice models, and the work of Davis et al. (2014) who optimize the assortment for the nested logit model. Implicit in this general approach is the caveat that one should fit the *right* parametric choice model to data before making predictions or decisions; this is a difficult problem since the implicit pre-specified structures in the parametric choice models might not be true in practice. There is a huge risk of model mis-specification, which will lead to inaccurate decision making downstream. Thus, there is a trade-off between specification and estimation: while a complex model can approximate a wider range of choice behaviors and hence may have smaller specification error, in return, it may have big estimation errors when estimated without sufficient data.

**Non-parametric choice models.** Non-parametric choice models consider distributions over all rankings. They have risen in popularity due to the increased availability of data; see Farias et al. (2013), van Ryzin and

<sup>1</sup> Somewhat related are cascade choice models (Aggarwal et al. 2008, Kempe and Mahdian 2008, Golrezaei et al. 2021), originally defined to model click probabilities in sponsored search ads. Cascade models capture a consumer going through items sequentially until choosing an item to buy or exiting the process.

Vulcano (2015), van Ryzin and Vulcano (2017), Haensel and Koole (2011), Jagabathula and Rusmevichientong (2017), Aouad et al. (2019), Honhon et al. (2012), and Rusmevichientong et al. (2010). Our work is most similar to that of Farias et al. (2013), van Ryzin and Vulcano (2015) and van Ryzin and Vulcano (2017), although all of them only use *offline* transaction data to estimate the best fitting choice model. Specifically, Farias et al. (2013) estimate the sparsest choice model consistent with the available transaction data; van Ryzin and Vulcano (2015) start from a parsimonious set of preferences, then iteratively add new preferences that increase the likelihood value of the data using a column generation based procedure; van Ryzin and Vulcano (2015) develop an efficient and easy-to-implement expectation maximization method that finds the non-parametric choice model with the highest likelihood.

Our work is different in one main aspect: we assume that the algorithm gets to determine the assortments offered to consumers. This allows the algorithm to acquire the most useful data, by querying the right assortments. As a result, our algorithm overcomes the identifiability issues that arise when one estimates the choice model using offline data. We highlight that instead, van Ryzin and Vulcano (2015) and Farias et al. (2013) aim to overcome this issue by imposing *assumptions* on the observed data. In van Ryzin and Vulcano (2015) and van Ryzin and Vulcano (2017), the authors impose that for each offer set  $S$  and product  $z$  in the set, there exists a preference ranking under which  $z$  ranks highest in  $S$ ; thus, they only consider items that are preferred the most by at least one type. In contrast, we consider all items and learn the set of the most popular items. Farias et al. (2013) impose the assumption that for every preference appearing in the population, there exists a query and an item in the data such that the item ranks the highest in the query only for that preference, and that the set of probabilities is linearly independent with respect to the integers. We do not need to make any such assumptions, because the DAG representation is unique given the transaction data.

There is a long line of work that aims to design algorithms identifying optimal or near-optimal revenue maximizing assortments, under the assumption that the model primitives are known. These studies are usually accompanied by a case study in which the model primitives are estimated from either real or synthetic data sets. The case studies are usually used to evaluate the proposed assortment planning algorithms; see, for example, Haensel and Koole (2011), Honhon et al. (2012), Jagabathula and Rusmevichientong (2017), Aouad et al. (2018, 2019), Feldman et al. (2019), Derakhshan et al. (2020) for some of the work that follows this approach for non-parametric choice models. To estimate the choice models, these approaches mostly use maximum likelihood estimators (MLE): the locally optimal solutions are obtained via expectation maximization (EM) algorithms. Notice that the estimation process in these works tends to be conducted under restrictive assumptions. While these assumptions are mainly imposed to ensure the tractability of the assortment planning problem, the assumptions can simplify the estimation process as well. For example, Feldman et al. (2019) assume that consumer types are derived from paths in a tree, where the set of preference lists consists of ordered nodes visited along each path in the tree. This assumption allows them to use MLE on

a domain of polynomial support size, to derive a fitted general tree representing the choice model while only considering linear paths on the tree. Honhon et al. (2012) assume that all the products can be mapped onto a hierarchical ordering system, such as branched, vertical or horizontal order; they then use a dynamic programming based algorithm to find the optimal assortment.

**Other uses of DAGs in non-parametric choice models.** In this work, one of our key contributions is the use of DAGs to represent non-parametric choice models. Our use of DAGs differs significantly from that in past work on choice models (Jagabathula and Vulcano 2018, Jagabathula et al. 2022). In the model presented by Jagabathula and Vulcano (2018), each customer type is characterized by a partial order over items, representing a collection of pairwise preference relations. This partial order is conveniently visualized using a DAG, with each node corresponding to an item. A directed edge from item  $a$  to item  $b$  indicates that item  $a$  is preferred over item  $b$  in the partial order. The full ranking of a customer is drawn randomly from a distribution specific to the customer's segment, consistent with the partial order. The customer then selects the highest-ranked item from their consideration set. The primary goal of the research in Jagabathula and Vulcano (2018) is to estimate this model using offline data, in contrast to the active learning approach in our work. Jagabathula et al. (2022) extend the DAG concept in Jagabathula and Vulcano (2018) to include promotions. Here, each item is represented by two nodes: one for its full price and another for its discounted version, thus optimizing promotion planning through this enhanced DAG framework.

Our use of DAGs as a modeling tool is fundamentally different from their uses, in that in our setting, the DAG is used to represent the choice model of all customer types at once, rather than just one customer type. That is, in our definition of DAGs, by labeling nodes with sets (rather than items), we manage to represent the entire choice model, rather than partial orders.

**Online learning for assortment planning.** The literature on online learning for assortment planning is also related to our work. In this literature, the goal is to learn the consumers' preferences from their actions, in order to identify the revenue-optimal assortments. This is mainly done for parametric choice models such as the multinomial logit, nested logit, Markov Chain, and cascade choice models as seen, for example, in Rusmevichientong et al. (2010), Sauré and Zeevi (2013), Agrawal et al. (2019), Chen et al. (2021), Niazadeh et al. (2020), Gallego and Lu (2021), Golrezaei et al. (2022). While these works mostly care about identifying the optimal assortment and minimizing regret, we focus on estimating the choice model itself. Furthermore, online learning has, to the best of our knowledge, not yet been studied for non-parametric choice models.

**Active learning.** Active learning, sometimes also known as “query learning” or “optimal experimental design” in statistics, is a field of machine learning in which the learning algorithm gets to *choose* the queries (unlabeled data), to be labeled by an oracle which knows the true labels. This control over the training data typically improves the performance of the algorithm with less training data. Within machine learning, the idea of active or online learning was introduced in the seminal papers of Angluin (1988) and Littlestone

(1988), in the context of learning a binary classifier. Naturally, combining offline or unlabeled data with data obtained online is frequently the best or most natural choice in practice; see, e.g., Cohn et al. (1996). Although active learning has been well studied in the literature (see Settles (2009) and Aggarwal et al. (2014) for surveys), to the best of our knowledge, active learning has not been used for learning choice models. Previously, several works in the literature have explored and shown the value of active learning compared to its “passive” counterpart. For example, Zheng and Padmanabhan (2006) developed a new active learning technique for an information acquisition problem in order to, for example, predict the probability of purchase and credit default rate. They demonstrated that the proposed method performs well empirically. Similarly, Aviv and Pazgal (2002) explored the value of proactively setting prices to impact the revenue.

A line of work similar to active learning is the dynamic sampling literature where one dynamically selects samples to learn parameters while optimizing the objective functions; see Shi et al. (2021), Zhang et al. (2020), Shin et al. (2018). Shi et al. (2021) dynamically allocate samples in a finite sampling budget to learn the system’s feasible alternatives efficiently in a feasibility determination problem appearing in many applications, such as call center design and hospital resource allocation. Shin et al. (2018) intelligently allocate samples in their simulation to minimize the probability of selecting a system that does not have the highest mean out of several competing alternatives (“systems”), when the probability distribution determining each system’s performance is unknown but can be learned from a limited number of samples they can obtain. However, none of these dynamic sampling approaches was done in the assortment planning or choice modeling setting.

### 3. Model

We use the following standard conventions.  $[k] = \{1, \dots, k\}$  is the set of the first  $k$  integers. Vectors are bold. When some quantity (such as a frequency/probability  $p$  or a set  $V$  etc.) has a ground truth value and an estimate by an algorithm, we use  $\hat{p}$  (or  $\hat{V}$  etc.) to distinguish the estimate from the ground truth.

#### 3.1. Choice Model

Let  $\mathcal{N} = \{1, 2, \dots, n\}$  be the universe of items and  $\Pi$  be the set of all possible rankings over  $\mathcal{N}$ . That is,  $|\Pi| = n!$ . We consider a non-parametric choice model, where each ranking  $\pi \in \Pi$  over  $\mathcal{N}$  represents the preferences of a type.  $\pi(i)$  is the  $i^{\text{th}}$  most preferred item by consumers of type  $\pi$ , and for any item  $z \in \mathcal{N}$ ,  $\pi^{-1}(z)$  is the rank/position of item  $z$  for consumers of type  $\pi$ . For each ranking  $\pi \in \Pi$ , let  $p(\pi)$  denote the probability of a consumer having the ranking  $\pi$ . Note that for some  $\pi \in \Pi$ , we can have  $p(\pi) = 0$ . When a consumer of type  $\pi$  is offered an assortment of items  $S \subseteq \mathcal{N}$ , she chooses her most preferred item among those in  $S$ ; that is, she chooses the item  $z \in S$  such that  $\pi^{-1}(z) < \pi^{-1}(x)$  for all  $x \in S, x \neq z$ . We write  $\Theta_z(S) = \{\pi \in \Pi \mid \pi^{-1}(z) < \pi^{-1}(x) \text{ for all } x \in S, x \neq z\}$  for the set of types  $\pi \in \Pi$  that choose item  $z \in S$ .



when the set  $S$  is offered. Then, for any set  $S$  and item  $z \in S$ , we define  $q_z(S)$  as the probability that a consumer chooses item  $z$  when the set  $S$  is offered. That is,

$$q_z(S) = p(\Theta_z(S)) = \sum_{\pi \in \Theta_z(S)} p(\pi).$$

### 3.2. Frequent vs. Rare Rankings, and a Generative Model

Types that only occur very rarely would need a very large number of samples to be estimated accurately. Thus, our primary goal is to estimate the DAG on types that occur sufficiently frequently. Specifically, in our model, there is a set  $\Pi_0^F$  of (at most)  $K$  *candidate frequent types*. In our *general type* model,  $\Pi_0^F$  can be chosen adversarially. Under the *random type* model,  $\Pi_0^F$  is chosen as a set of  $K$  i.i.d. uniformly drawn rankings from the set of all rankings.<sup>2</sup> All of our correctness results for inferring types hold in the general type model, while the sample efficiency results hold under the random type model.

An adversary chooses the actual *frequent types*  $\Pi^F \subseteq \Pi_0^F$  from the set of candidate frequent types; all remaining types with positive probability,  $\Pi^R = \{\pi \in \Pi \mid p(\pi) > 0, \pi \notin \Pi^F\}$ , are called *rare types*. The adversary then assigns probabilities  $p(\pi)$  to each type/ranking  $\pi \in \Pi$ , subject to the following constraints:

1. The frequent types each appear at least a  $\kappa$  fraction of the time, i.e.,  $p(\pi) \geq \kappa$  for all  $\pi \in \Pi^F$ .
2. The rare types appear less than  $\kappa$  fraction of the time, and cumulatively, they appear at most a  $\rho$  fraction of the time, i.e.,  $p(\pi) < \kappa$  for all rare types  $\pi \in \Pi^R$  and  $\sum_{\pi \in \Pi^R} p(\pi) \leq \rho$ .
3. The probabilities define a distribution, i.e.,  $p(\pi) \geq 0$  for all  $\pi$  and  $\sum_{\pi \in \Pi} p(\pi) = 1$ .

When the candidate frequent rankings  $\Pi_0^F$  can be adversarial, we call the model with parameters  $\kappa \in (0, 1)$  and  $\rho \in [0, 1)$  the  $(\kappa, \rho)$ -General Model; when  $\Pi_0^F$  is random, we refer to the model as the  $(\kappa, \rho)$ -Random Model. We note that our random model captures a scenario where consumer types are very heterogeneous.

Notice that due to the lower bound of  $\kappa$  on probabilities in  $\Pi^F$ , we have that  $|\Pi^F| \leq \frac{1}{\kappa}$ . Intuitively, we can think of  $\Pi^F$  as a set of main types in the choice model and  $\Pi^R$  as the noise. An algorithm's goal will then be to accurately infer the rankings in  $\Pi^F$  and their probabilities; the larger the combined probability  $\rho$  of the rare types (i.e., the noise), the less accurate the estimates will become.

### 3.3. Active Learning of the Choice Model

We are interested in *actively learning* the choice model in a setting where consumers' types are not observable. In an active learning framework, the algorithm gets to decide — based on all past observed choices — which subset/assortment  $S \subseteq \mathcal{N}$  to offer to the next consumer. The algorithm does so without knowing the next consumer's ranking/type, which is drawn from the choice model, independently of all past consumers' types. That is, the consumer is of type  $\pi \in \Pi$  with probability  $p(\pi)$ . Upon being presented with the assortment  $S$ , the consumer chooses her most preferred item within the set  $S$  according to her ranking  $\pi$ . For

<sup>2</sup> If there are duplicates among the random  $K$  rankings, then  $|\Pi_0^F| < K$ .

simplicity, we assume that a consumer always chooses an item when the assortment offered is non-empty; that is, no type has a no-purchase option.<sup>3</sup> We emphasize that due to practical consideration (consider an online retail site in which each consumer purchases items much less frequently than the overall rate of transactions), we assume that each consumer can only be queried once. This prevents an algorithm from immediately collecting detailed information about a type via multiple queries.

The algorithm’s goal is then to learn, with probability at least  $1 - \delta$ , the top  $n_0$  positions of all frequent rankings (i.e.,  $(\pi(1), \pi(2), \dots, \pi(n_0)), \pi \in \Pi^F$ ) and their corresponding probabilities within accuracy  $\epsilon$  (plus an error term depending on  $\rho$ ), using a number of queries which is polynomial in  $n, 1/\kappa$  and  $1/\epsilon$ , and  $1/\delta$ . (We use the terms “samples” and “queries” interchangeably throughout the paper.) Here,  $n_0 = \alpha n$  for some constant  $\alpha \in (0, 1)$ . Note that the algorithm need not learn the rare types; after all, it could take a large number of queries to even observe a single sample from the rare types. We also are only interested in learning the  $n_0$  most popular items in the frequent rankings. Knowing the most popular items could help decision makers with inventory planning and product design decisions, to name a few. Furthermore, learning the top  $n_0$  items can be justified from both practical and technical perspectives. Practically speaking, consumers are often unsure about their preferences for the non-top items in their ranking; hence, they can be inconsistent in choosing among the set of items at the bottom of their ranking (Chernev (2006), Goldin and Reck (2015)). In other words, even if the items appearing in low positions were learned, the results might not be very reliable. Moreover, the items at the end of a consumer’s ranking are only purchased when none of her top items are available in the offered sets. Thus, when we see an item in one of the bottom positions of a type being purchased in the transaction data, this data point is likely the result of a different consumer type which ranks the particular item higher. From a technical perspective, as shown in Theorem 3, as  $n_0$  gets larger, distinguishing different types of consumers in order to learn their rankings requires a large number of queries. Given the cost of querying, one would like to avoid having large values for  $n_0$ , i.e., there is a tradeoff between using few queries vs. learning the bottom fraction of items of each type. A more in-depth discussion of the implications of approximation and truncation can be found in Section 8.

#### 4. Indistinguishable Pairs of Rankings

In this section, we show that when the set of frequent rankings  $\Pi^F$  contains an *indistinguishable* pair of rankings (in the sense of the following definition), it is information-theoretically impossible to discover the set of types  $\Pi^F$  uniquely. This motivates our choice to use DAGs as a representation of the types — in a sense, they extract the most information that *can* be learned.

**DEFINITION 1 (INDISTINGUISHABILITY).** Two rankings  $\pi$  and  $\pi'$  are  $i$ -indistinguishable for some  $2 \leq i \leq n - 2$  if and only if they satisfy the following three conditions:

<sup>3</sup> When one allows for a no-purchase option in rankings, learning rankings over items that appear after the no-purchase option becomes impossible. For example, assume that, for a consumer type, the no-purchase option is the second most popular item. Then, any active learning algorithm can only learn the most popular item of this consumer type.

1. the set of items in the first  $i$  position of  $\pi$  and  $\pi'$  is the same, i.e.,  $\{\pi(j)|j \in [i]\} = \{\pi'(j)|j \in [i]\}$ ,
2. at least one item among the top  $i$  items has different positions in  $\pi$  and  $\pi'$ , i.e., for at least one  $j \in [i]$ , we have  $\pi(j) \neq \pi'(j)$ , and
3. at least one item among the bottom  $n - i$  items has different positions in  $\pi$  and  $\pi'$ , i.e., for at least one  $j \in \{i + 1, \dots, n\}$ , we have  $\pi(j) \neq \pi'(j)$ .

Two rankings are *indistinguishable* if they are  $i$ -indistinguishable for some  $2 \leq i \leq n - 2$ .

The following is the main result of this section.

**THEOREM 1 (Impossibility Result).** *Suppose that the set of frequent rankings  $\Pi^F$  contains two rankings  $\pi$  and  $\pi'$  that are indistinguishable. Then, it is information-theoretically impossible to discover the set of types  $\Pi^F$  uniquely.*

**Proof Sketch.** Given permutations  $\pi$  and  $\pi'$  that are  $i$ -indistinguishable for  $2 \leq i \leq n - 2$ , where  $n$  denotes the number of items, we define  $\bar{\pi}$  and  $\bar{\pi}'$  as follows:  $\bar{\pi}$  is identical to  $\pi$  in the first  $i$  positions and identical to  $\pi'$  in the remaining positions.  $\bar{\pi}'$  is identical to  $\pi'$  in the first  $i$  positions and identical to  $\pi$  in the remaining positions. Given probabilities  $p$ , we then construct an alternate choice model  $\bar{p}(\cdot)$  where  $\bar{\pi}, \bar{\pi}'$  are assigned probabilities similar to  $\pi$  and  $\pi'$ , respectively, with slight adjustments. Specifically,  $\bar{p}(\bar{\pi}) = \bar{p}(\bar{\pi}') = p(\pi)$ ,  $\bar{p}(\pi') = p(\pi') - p(\pi)$ , and  $\bar{p}(\pi) = 0$ . For all other permutations  $\pi'' \in \Pi$ ,  $\bar{p}(\pi'') = p(\pi'')$ . Using a coupling argument, detailed in Appendix A.1, one can then show that no algorithm can distinguish between the choice models  $p$  and  $\bar{p}$ . ■

We remark that even if the frequent types are drawn i.i.d. uniformly, indistinguishable pairs are likely to occur — in particular, assuming the random model for frequent types does not obviate the need for DAG representations. To see this, consider  $i = 2$ . Given a ranking  $\pi$ , a uniformly random ranking starts with  $(\pi(2), \pi(1))$  with probability  $\frac{1}{n(n-1)}$ , so it is 2-indistinguishable from  $\pi$  with probability  $\frac{1}{n(n-1)} \cdot (1 - 1/(n-2)!) \geq \frac{1}{n^2}$  (for  $n \geq 4$ , which is necessary for indistinguishability). By a Birthday Paradox argument, a 2-indistinguishable pair of types will occur with constant probability when  $K = \Omega(n)$ , and with high probability when  $K = \omega(n)$ . (Recall that  $K$  is an upper bound on the number of frequent types.) Thus, as the number of types will typically exceed the number of items, indistinguishable pairs must be accounted for, even in our random model.

## 5. DAG Representation of a Choice Model

Theorem 1 shows that when there are indistinguishable pairs of (frequent) rankings, no algorithm can learn the choice model. However, even if the set of frequent rankings  $\Pi^F$  contains pairs of indistinguishable rankings, we would still like to recover as much information as possible. We therefore introduce the Directed Acyclic Graph (DAG) representation for a choice model, which effectively encodes the maximum information that can be deduced from a sampling process, in a sense we will elaborate on in Section 6.1.

The motivation for the DAG model arises from the proof of Theorem 1. Consider two  $i$ -indistinguishable rankings  $\pi, \pi'$ . We cannot tell apart a world in which just these two types are present, or some/all of the probability lies on types combining the ranking of the first  $i$  items according to  $\pi$  with the ranking of the remaining  $n - i$  items according to  $\pi'$  (and vice versa). Let  $A = \{\pi(1), \dots, \pi(i)\} = \{\pi'(1), \dots, \pi'(i)\}$  denote the common set of  $i$  items ranked first (in different orders) by  $\pi, \pi'$ . Then, a sampling-based algorithm *can* infer the probabilities with which  $A$  is ranked according to  $\pi$  and according to  $\pi'$ , and similarly for  $\mathcal{N} \setminus A$ ; it just cannot infer how these rankings are “combined.” We can consider this as the rankings  $\pi, \pi'$  “merging” after position  $i$ , because they both have the set  $A$  in the first  $i$  positions; subsequently, they split again, but the “merge point” corresponds to an indistinguishability. This intuition generalizes to more complex similarities and differences between types, and leads to our definition of the DAG representation. We will later see in Figure 2 that  $\Pi^F$  and  $\bar{\Pi}^F$  in Theorem 1, with their corresponding types’ probability, have the same DAG representation.

### 5.1. Edges and Prefixes, and their Probabilities in the DAG Representation

The DAG representation relies on the notion of *prefixes* in the choice model. We begin by defining the following nomenclature for prefixes and sets of prefixes.

**DEFINITION 2 (PREFIXES, EDGES, AND THEIR PROBABILITIES).** 1. A *prefix* of size  $j$  of a ranking  $\pi$  is the set of the top (most preferred)  $j$  items in type  $\pi$ , i.e.,  $\{\pi(j') \mid j' = 1, \dots, j\}$ .  
2. The probability of a prefix  $A$  of size  $j$  is the probability that the set of the top- $j$  items of a random consumer type equals  $A$ , i.e.,

$$p(A) = \sum_{\pi: \{\pi(j') \mid j'=1, \dots, j\} = A} p(\pi). \quad (1)$$

3. For  $z \notin A$ , we write  $\Theta_{A \oplus z} = \{\pi \mid \{\pi(1), \pi(2), \dots, \pi(j)\} = A, \pi(j+1) = z, p(\pi) > 0\}$  for the set of types/rankings of positive probability that have the set  $A$  in the first  $j$  positions (in any order) and the item  $z$  in position  $j+1$ . We then write

$$e_{A \oplus z} = \sum_{\pi \in \Theta_{A \oplus z}} p(\pi) \quad (2)$$

for the combined probability of such types/rankings. We refer to  $e_{A \oplus z}$  as the *edge probability* between nodes  $A$  and  $A \cup \{z\}$ .

The following observations follow immediately from the definitions:

**PROPOSITION 1.** 1. For fixed  $z$ , the sets  $\Theta_{A \oplus z}$  of types are disjoint, i.e.,  $\Theta_{A \oplus z} \cap \Theta_{A' \oplus z} = \emptyset$ , for any  $A, A'$  with  $A \neq A'$ .

2. The prefix probabilities can be expressed as follows:

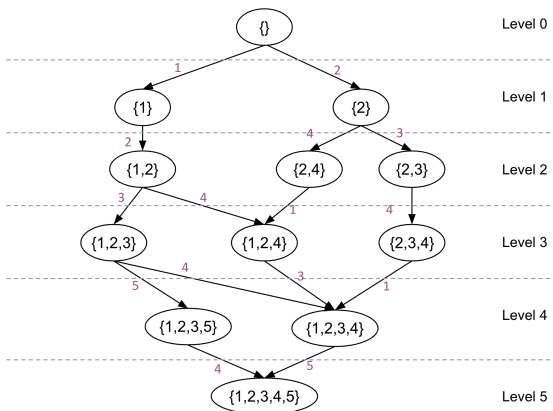
$$p(A) = \sum_{z \notin A} e_{A \oplus z} = \sum_{z \in A} e_{A \setminus \{z\} \oplus z}, \quad (3)$$

where the first representation applies only when  $A$  is not the set of all the items, and the second representation applies only for non-empty  $A$ .

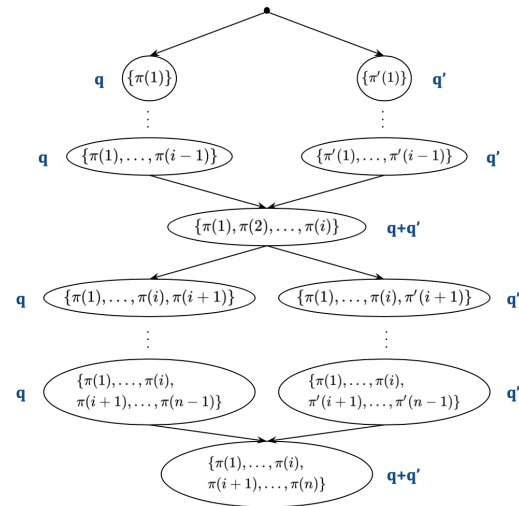
The second part of the proposition suggests interpreting the  $e_{A \oplus z}$  as transition probabilities between prefixes, and motivates the nomenclature of “edges”. Indeed, prefixes and edges between them exactly form the primitives of the DAG representation.

## 5.2. The DAG Representation

Utilizing the notions of prefixes and edges, we now define the DAG representation of a non-parametric choice model.



**Figure 1** The DAG corresponding to  $(\Pi, p)$  with  
 $p((1, 2, 3, 4, 5)) = p((1, 2, 3, 5, 4)) =$   
 $p((1, 2, 4, 3, 5)) = p((2, 3, 4, 1, 5)) =$   
 $p((2, 4, 1, 3, 5)) = 0.2.$



**Figure 2** The DAG corresponding to the indistinguishable pairs of rankings from Theorem 1.

**DEFINITION 3 (DAG REPRESENTATION OF  $(\Pi, p)$ ).** Let  $V = \{A \mid p(A) > 0\}$  be the set of all prefixes  $A$  with positive probability, and  $E = \{(A, A \cup \{z\}) \mid e_{A \oplus z} > 0\}$  the set of all edges with positive probability.

The Directed Acyclic Graph (DAG) representation of the choice model  $(\Pi, p)$ , denoted by  $G = (V, E, (p(A))_{A \in V}, (e_{A \oplus z})_{(A, A \cup \{z\}) \in E})$ , has node set  $V$  and edge set  $E$ . The probabilities of nodes  $A$  and edges  $(A, A \cup \{z\})$  are  $p(A)$  and  $e_{A \oplus z}$ , respectively. We refer to the prefixes  $A$  with  $|A| = j$  as *level  $j$*  of the graph<sup>4</sup>, and to  $z$  as the *label* of the edge  $(A, A \cup \{z\})$ .

<sup>4</sup> Note that edges exist solely between consecutive levels.

Two DAG representations  $G = (V, E, \mathbf{p}, \mathbf{e})$  and  $G' = (V', E', \mathbf{p}', \mathbf{e}')$  are *identical* if and only if  $V = V'$ ,  $E = E'$ ,  $p(A) = p'(A)$  for all  $A \in V$ , and  $e_{A \oplus z} = e'_{A \oplus z}$  for all  $(A, A \cup \{z\}) \in E$ .

An example DAG illustrating this definition is shown in Figure 1, for the choice model  $(\Pi, \mathbf{p})$  whose types with positive probability are  $\{(1, 2, 3, 4, 5), (1, 2, 3, 5, 4), (1, 2, 4, 3, 5), (2, 3, 4, 1, 5), (2, 4, 1, 3, 5)\}$ , each with probability 0.2. In this example, the nodes on level one are  $\{1\}$  and  $\{2\}$  (the top elements under  $\Pi$ ), the nodes on level two are  $\{1, 2\}$ ,  $\{2, 3\}$ , and  $\{2, 4\}$  (the top pairs of elements under  $\Pi$ ), and the nodes on level three are  $\{1, 2, 3\}$ ,  $\{1, 2, 4\}$ , and  $\{2, 3, 4\}$  (the top triples). This example also shows that multiple edges can have the same label. For example, both the edge  $(\{1\}, \{1, 2\})$  and  $(\emptyset, \{2\})$  are labeled 2. For a second example, see Figure 2, which shows the DAG representation of  $\Pi^F$  and  $\bar{\Pi}^F$  from Theorem 1. There, it can be observed that  $\Pi^F$  and  $\bar{\Pi}^F$  have the same DAG representation, and that the DAG merges at level  $i$  and splits at level  $i + 1$ .

We finish this section by defining the notion of truncated DAGs, which are useful for describing the intermediate stages of the DAG construction algorithm.

**DEFINITION 4 (TRUNCATED DAG).** For any DAG model  $G = (V, E, \mathbf{p}, \mathbf{e})$ , given an integer  $j \in \{0, 1, \dots, n\}$ , we let  $G_j$  denote the DAG (with associated probabilities) restricted to levels  $0, \dots, j$ . Formally, we let  $V_j = \{A \in V \mid |A| \leq j\}$  be the set of prefixes in  $V$  of at most  $j$  items,  $E_j = E[V_j \times V_j]$  the set of edges induced by these prefixes, and  $\mathbf{p}_j = (p(A))_{A \in V_j}$  and  $\mathbf{e}_j = (e_{A \oplus z})_{A \in V_j, A \cup \{z\} \in V_j}$ . Then,  $G_j = (V_j, E_j, \mathbf{p}_j, \mathbf{e}_j)$ .

## 6. DAGs and Choice Probabilities

We first show that the DAG representation captures all the information inherent in the choice probabilities, in the sense that the choice probabilities can be fully (and efficiently) recovered from the corresponding DAG representation. In turn, given oracle access to the (exact) choice probabilities, the DAG representation of a choice model can be constructed in time polynomial in the number of types. The proof of the latter result also serves as an easy warm-up, illustrating the central underlying ideas that we significantly refine to adaptively deal with sampled choice frequencies, rather than exact choice probabilities.

### 6.1. Computing the Choice Probabilities from a given DAG

Given an item set  $S$  and item  $z \in S$ , we would like to compute the probability  $q_z(S)$  of a random type choosing  $z$  when offered the set  $S$ . The calculation of choice probabilities is captured by the following proposition.

**PROPOSITION 2 (Computing Choice Probabilities from a DAG).** Let  $(\Pi, \mathbf{p})$  be a choice model, and  $(V, E, \mathbf{p}, \mathbf{e})$  a DAG representation of  $(\Pi, \mathbf{p})$ . Let  $S \subseteq \mathcal{N}$  be an item set and  $z \in S$ . Then, the set of types choosing  $z$  from  $S$  and its associated probability are characterized by the following:

$$\Theta_z(S) = \bigcup_{A: A \cap S = \emptyset} \Theta_{A \oplus z} \quad q_z(S) = \sum_{A: A \cap S = \emptyset} e_{A \oplus z}.$$

Proposition 2, which is shown in Appendix A.2, implies a straightforward linear-time algorithm for computing  $q_z(S)$ : iterate through all prefixes  $A \in V$ , and if  $A \cap S = \emptyset$ , then add  $e_{A \oplus z}$  (which might be 0, if the edge does not exist) to the total.

## 6.2. Constructing the DAG with an Exact Choice Probability Oracle

In this section, we show that when an algorithm has access to the exact choice probabilities — as opposed to empirical and noisy choice frequencies — for arbitrary sets and items, it can reconstruct the DAG efficiently. We model such access by an oracle which, given  $S$  and  $z$ , will correctly return  $q_z(S)$ .

The algorithm (Algorithm 1) constructs the DAG level by level. Recall that throughout the paper, we denote quantities which the algorithm computes by a hat, to distinguish them from the (possibly different) ground truth values. The algorithm starts with level 0, composed of only the node  $\emptyset$  with  $\hat{p}(\emptyset) = 1$ . In each iteration  $j = 0, 1, \dots, n-1$ , for each node  $A$  on level  $j$ , it determines all outgoing edges  $(A, A \cup \{z\})$  of positive probability using Lemma 1 below. Then, level  $j+1$  is composed of all nodes  $A \cup \{z\}$  that were the head of at least one of these edges. The probability associated with a node  $A$  on level  $j+1$  can be obtained from Proposition 1 as  $p(A) = \sum_{z \in A} e_{A \setminus \{z\} \oplus z}$ .

**LEMMA 1.** *For any prefix  $A$  and  $z \notin A$ , we have  $e_{A \oplus z} = q_z(\mathcal{N} \setminus A) - \sum_{A' \subsetneq A} e_{A' \oplus z}$ .*

**Proof.** Proposition 2 with  $S = \mathcal{N} \setminus A$  gives  $q_z(\mathcal{N} \setminus A) = \sum_{A': A' \cap (\mathcal{N} \setminus A) = \emptyset} e_{A' \oplus z} = \sum_{A' \subsetneq A} e_{A' \oplus z}$ . Solving for  $e_{A \oplus z}$  proves the lemma. ■

---

### Algorithm 1: Constructing the Exact DAG using Oracle Access to Exact Choice Probabilities

---

**Input:** An exact oracle for the choice probabilities of a non-parametric choice model  $(\Pi, \mathbf{p})$ .

**Output:** The (exact) DAG representation  $G$  of the choice model and its associated probabilities.

```

1 Initialize  $G_0 = (\{\emptyset\}, \emptyset)$  with only one node for the empty set, and no edges. Set  $\hat{p}(\emptyset) = 1$ .
2 for  $j = 0, \dots, n-1$  do
3   Initialize  $V_{j+1}$  and  $E_{j+1}$  as empty sets.
4   for each set  $A$  on level  $j$  of  $G_j$  do
5     for each item  $z \notin A$  do
6       Set  $\hat{e}_{A \oplus z} = q_z(\mathcal{N} \setminus A) - \sum_{A' \subsetneq A} \hat{e}_{A' \oplus z}$ ;           /* Compute the edge probabilities */
7       if  $\hat{e}_{A \oplus z} > 0$  then
8         if node  $A \cup \{z\}$  is not in  $V_{j+1}$  then
9           Add node  $A \cup \{z\}$  to  $V_{j+1}$ . Set  $\hat{p}(A \cup \{z\}) = 0$ .
10          Set  $\hat{p}(A \cup \{z\}) = \hat{p}(A \cup \{z\}) + \hat{e}_{A \oplus z}$ .
11          Add a directed edge from  $A$  to  $A \cup \{z\}$  in  $E_{j+1}$ , labeled with  $z$ , and with probability  $\hat{e}_{A \oplus z}$ 
12          .

```

---

The performance of Algorithm 1 is captured by the following theorem, whose proof is in Appendix A.3.

**THEOREM 2 (Constructing the Exact DAG using Exact Choice Probabilities).** *Let  $(\Pi, \mathbf{p})$  be the underlying choice model with  $T$  types, and assume that Algorithm 1 has exact oracle access to the choice probabilities. Then, Algorithm 1 runs in time  $O(n^2T)$  and outputs a correct DAG representation of  $(\Pi, \mathbf{p})$ .*

## 7. Active Learning of a DAG using Estimated Choice Frequencies

We presented Algorithm 1 and its analysis under the assumption of exact access to choice probabilities. In reality, information about consumers' choice behavior will virtually always be inferred from observations, and as such be based on *empirical frequencies*, which will differ subtly from exact choice probabilities.

One can of course still use Algorithm 1 to construct a DAG, using empirical estimates of the quantities  $q_z(\mathcal{N} \setminus A)$ . However, this will lead to corresponding misestimates of the quantities  $e_{A \oplus z}$ . Importantly, the key equation from Lemma 1 for computation of edge probabilities,  $e_{A \oplus z} = q_z(\mathcal{N} \setminus A) - \sum_{A' \subsetneq A} e_{A' \oplus z}$ , uses previously computed edge probability estimates in later iterations. As such, it is susceptible to an accumulation of errors, which might amplify errors. That this can indeed occur is illustrated in the following example, depicted in Figure 3:

**EXAMPLE 1.** Consider a scenario with eight items and the following frequent rankings:

$$\Pi = \{(1, 2, 3, 4, 5, 6, 7, 8), (1, 2, 3, 4, 5, 6, 8, 7), (1, 2, 3, 4, 7, 5, 6, 8), (1, 2, 3, 5, 7, 4, 6, 8), (1, 2, 3, 6, 7, 4, 5, 8), (1, 7, 2, 3, 4, 5, 6, 8), (2, 7, 1, 3, 4, 5, 6, 8), (3, 7, 1, 2, 4, 5, 6, 8), (7, 1, 2, 3, 4, 5, 6, 8)\}.$$

The goal is to estimate the edge probability  $e_{\{1,2,3,4,5,6\} \oplus 7}$ , corresponding to the bolded node and edge on the 7<sup>th</sup> level of the DAG, as highlighted in purple in Figure 3. The proper subsets  $A' \subsetneq A$  with an outgoing edge labeled  $z = 7$  are  $\{1, 2, 3, 4\}, \{1, 2, 3, 5\}, \{1, 2, 3, 6\}, \{1\}, \{2\}, \{3\}, \emptyset$ , highlighted in blue.

To compute the edge probability  $e_{\{1,2,3,4,5,6\} \oplus 7} = e_{A \oplus z}$ , one can repeatedly apply Lemma 1 to all terms of the form  $e_{A' \oplus z}$  and rearrange, obtaining that

$$\begin{aligned} \sum_{A' \subsetneq A} e_{A' \oplus z} &= e_{\{1,2,3,4\} \oplus 7} + e_{\{1,2,3,5\} \oplus 7} + e_{\{1,2,3,6\} \oplus 7} + e_{\{1\} \oplus 7} + e_{\{2\} \oplus 7} + e_{\{3\} \oplus 7} + e_{\emptyset \oplus 7} \\ &= q_7(\mathcal{N} \setminus \{1, 2, 3, 4\}) + q_7(\mathcal{N} \setminus \{1, 2, 3, 5\}) + q_7(\mathcal{N} \setminus \{1, 2, 3, 6\}) \\ &\quad - 2q_7(\mathcal{N} \setminus \{1\}) - 2q_7(\mathcal{N} \setminus \{2\}) - 2q_7(\mathcal{N} \setminus \{3\}) + 4q_7(\mathcal{N} \setminus \emptyset). \end{aligned} \tag{4}$$

Observe that the coefficients grow exponentially in the number of times a set occurred in the expansion steps, as observable in the coefficient  $q_7(\mathcal{N} \setminus \emptyset)$  with the largest absolute value, i.e., 4. This exponential growth pattern persists for a generalization of this example to instances with  $(3k - 1)$  items and  $3k$  types, where the biggest coefficient has an absolute value of  $2^{k-1}$ .

Having exponentially large coefficients is not appealing, because they exponentially amplify any estimation errors due to rare rankings. Counteracting such estimation errors with improved estimation accuracy would then require exponentially many samples. Our main technical contribution is a more careful way to select the queries to use, based on approximate solutions to Set Cover instances that we will introduce



shortly. This approach guarantees that the resulting linear system has a good condition number. While the number of distinct queries required is exponential in the size of the Set Cover solution, whenever this solution has only logarithmic size, the resulting number of queries is polynomial. In particular, this is the case under the random model.

## 7.1. The Active Learning Algorithm

In order to avoid the possibly exponential error accumulation, our adaptive algorithm ALGDAG— given as Algorithm 2 — modifies Algorithm 1 in three ways:

1. To compute the edge probabilities  $e_{A \oplus z}$ , rather than using Lemma 1, ALGDAG uses a more complex new algorithm ALGIE (Algorithm 3) based on the Inclusion-Exclusion Principle. This algorithm internally uses random samples from the population for estimating choice frequencies, and thus must be given error parameters  $(\epsilon, \delta)$ . It will adjust the number of samples to ensure that with probability at least  $1 - \delta$ , the estimate is accurate to within error  $\epsilon$ .
2. Rather than including all edges whose estimated probability is positive, ALGDAG only includes edges whose estimated probability is at least  $\kappa/2$ ; i.e., it focuses on edges with sufficiently high probability of corresponding types. The reason for excluding edges with small probability estimates is that their estimates are likely very inaccurate unless a large number of samples is used. As a result, ALGDAG produces a DAG only for the frequent types.
3. ALGDAG only builds the first  $n_0 = \alpha \cdot n$  levels of the DAG, for a user-specified parameter  $\alpha$ . The reason is that again, estimates for the lower levels tend to become less accurate.

Apart from these modifications, ALGDAG is identical to Algorithm 1. It obtains as input the parameters  $\alpha, \epsilon, \delta$ , in addition to the parameters  $\kappa, \rho$  of the choice model (or a valid lower bound thereof).

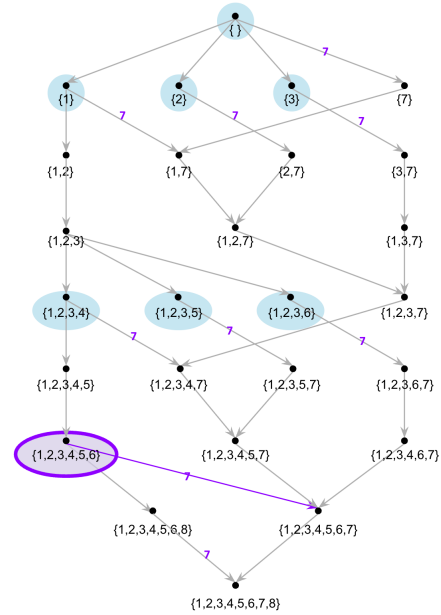


Figure 3: The DAG corresponding to Example 1. The rankings corresponding to the blue nodes interfere when trying to estimate the probability of the ranking corresponding to the purple node and edge. That is, the blue nodes represent the prefixes  $A' \subsetneq A = \{1, 2, 3, 4, 5, 6\}$  in Lemma 1.

**Algorithm 2:** Adaptive DAG Ranking Algorithm (ALGDAG)**Input:**  $\alpha$ ,  $\epsilon$ , and  $\delta$ .**Output:** A DAG representation  $(\hat{G}_{n_0}, \hat{p}, \hat{e})$  estimating the top  $n_0 = \alpha n$  items in the set of rankings  $\Pi^F$  with their associated probabilities

```

1 Initialize  $\hat{G}_0 = (\{\emptyset\}, \emptyset)$  as a graph with only one node for the empty set, and no edges. Set  $\hat{p}(\emptyset) = 1$ .
2 for  $j = 0, \dots, n_0 - 1$  do
3   Initialize  $\hat{V}_{j+1}$  and  $\hat{E}_{j+1}$  as empty sets.
4   for each set  $A$  on level  $j$  of  $\hat{G}_j$  do
5     for each item  $z \notin A$  do
6       Set  $\hat{e}_{A \oplus z}$  to  $\text{ALGIE}(\hat{G}_j, A, z; \alpha, \epsilon', \delta')$  with  $\epsilon' = \min(\epsilon/n_0, \kappa/4)$  and  $\delta' = \frac{\delta}{\alpha n^2 K}$ .
7       if  $\hat{e}_{A \oplus z} \geq \kappa/2$  then
8         if node  $A \cup \{z\}$  is not in  $\hat{V}_{j+1}$  then
9           Add node  $A \cup \{z\}$  to  $\hat{V}_{j+1}$ . Set  $\hat{p}(A \cup \{z\}) = 0$ .
10          Set  $\hat{p}(A \cup \{z\}) = \hat{p}(A \cup \{z\}) + \hat{e}_{A \oplus z}$ .
11          Add a directed edge from  $A$  to  $A \cup \{z\}$  in  $\hat{E}_{j+1}$ , labeled with  $z$ , and with probability  $\hat{e}_{A \oplus z}$ .
```

**7.2. The Inclusion-Exclusion Approach**

As we observed in Example 1 with a coefficient of 4 — and asserted to hold more generally — applying Lemma 1 repeatedly in the straightforward way can ultimately express the edge probabilities as a linear combination of choice frequencies in which the coefficients may grow exponentially. This exponential growth amplifies small estimation errors, and would thus require extremely accurate frequency estimates. It is desirable to instead express the edge probabilities as a linear combination of choice frequencies in a way that keeps the coefficients bounded. We will use the Inclusion-Exclusion formula to produce a linear combination in which all coefficients are  $\pm 1$ .

To motivate the approach, we revisit Example 1, depicted in Figure 3. There, the goal was to estimate the probability of the edge labeled 7 emanating from the purple node  $\{1, 2, 3, 4, 5, 6\}$ . To do so, we considered all nodes with outgoing edges labeled 7 which are ancestors of the node  $\{1, 2, 3, 4, 5, 6\}$ . These are the blue nodes  $\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2, 3, 4\}, \{1, 2, 3, 5\}, \{1, 2, 3, 6\}$ . We observe that each type corresponding to one of these prefixes prefers item 7 over at least one of  $\{5, 6\}$  (in addition to item 8, which 7 is preferred over by all ancestors of  $\{1, 2, 3, 4, 5, 6\}$ , by virtue of not being in the set  $\{1, 2, 3, 4, 5, 6\}$ ). As an example, all types corresponding to  $\{1, 2, 3, 5\}$  prefer both 7 and 8 over 6; see Figure 3. This observation “almost” allows us to write  $\sum_{A' \subsetneq \{1, 2, 3, 4, 5, 6\}} e_{A' \oplus z} = q_7(\{6, 7, 8\}) + q_7(\{5, 7, 8\})$ . The “almost” is because simply adding these two probabilities double-counts types who prefer 7 over *both* 5 and 6 (in addition to 8). To correct for the double-counting, we subtract the probability of double-counted types, which is  $q_7(\{5, 6, 7, 8\})$ . We thus obtain that  $\sum_{A' \subsetneq \{1, 2, 3, 4, 5, 6\}} e_{A' \oplus z} = q_7(\{6, 7, 8\}) + q_7(\{5, 7, 8\}) - q_7(\{5, 6, 7, 8\})$ .

Notice that in this formula, all coefficients are  $\pm 1$ . Inspecting the sets  $\{6, 7, 8\}, \{5, 7, 8\}$  more closely, we notice that their complements  $(\{1, 2, 3, 4, 5\}, \{1, 2, 3, 4, 6\})$  are direct predecessors of  $\{1, 2, 3, 4, 5, 6\}$ , and

each of the ancestors of interest (blue nodes) is also an ancestor of at least one of  $\{1, 2, 3, 4, 5\}, \{1, 2, 3, 4, 6\}$ . In this sense, these two sets *cover* all blues nodes, and this notion lies at the heart of our improved estimation approach. Formally, we define the following:

**DEFINITION 5 (SET COVER).** Fix a set  $A$  and element  $z \notin A$ . Let

$$\mathcal{S}^-(A) = \{A' \subsetneq A \mid |A'| = |A| - 1\}$$

denote the collection of subsets of  $A$  with size  $|A| - 1$ . Let  $\mathcal{P}(A, z) = \{A' \subsetneq A \mid e_{A' \oplus z} > 0\}$  be the set of all prefixes  $A' \subsetneq A$  which have an outgoing edge to the node  $A' \cup \{z\}$ , and let  $\mathcal{C} \subseteq \mathcal{P}(A, z)$ .

We say that  $\mathcal{C}$  is a *set cover* of  $\mathcal{P}$  if (1)  $\mathcal{C} \subseteq \mathcal{S}^-(A)$ , and (2) for each set  $A' \in \mathcal{P}$ , there exists a set  $C \in \mathcal{C}$  with  $A' \subseteq C$ , i.e., each set in  $\mathcal{P}$  is contained in some set in  $\mathcal{C}$ .

Definition 5 is closely related to the standard notion of set covers in algorithm design, a fact we elaborate on more in Appendix B, and which we exploit in our algorithm design.

When  $\mathcal{C}$  is a set cover of  $\mathcal{P}(A, z)$ , the Inclusion-Exclusion formula we previously saw by example generalizes as follows:

**PROPOSITION 3 (Inclusion-Exclusion with Exact Probabilities).** Fix a set  $A$  and item  $z \notin A$ , and let  $\mathcal{C}$  be a set cover of  $\mathcal{P}(A, z)$  using  $\mathcal{S}^-(A)$ . Then,

$$e_{A \oplus z} = q_z(\mathcal{N} \setminus A) - \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C).$$

Proposition 3, which is proven in Appendix C, serves as the central motivation for the Inclusion-Exclusion formula used in our algorithm ALGIE (Algorithm 3).

Because the right-hand side of the formula in Proposition 3, which is used to calculate the estimated edge probabilities, contains a number of terms exponential<sup>5</sup> in  $|\mathcal{C}|$ , it is desirable to make this set cover  $\mathcal{C}$  as small as possible. Unfortunately, as we show in Appendix B, our Set Cover problem subsumes the standard SET COVER problem, which is well known to be NP-hard. Thus, minimizing  $|\mathcal{C}|$  exactly is unlikely to be possible in polynomial time. Instead, we use the greedy algorithm, which is known (see, e.g., (Kleinberg and Tardos 2006)) to offer an approximation guarantee logarithmic in the number of items.

The greedy algorithm for SET COVER, denoted by GREEDYMINCOVER( $\mathcal{P}, \mathcal{S}^-(A)$ ), repeatedly selects a set from  $\mathcal{S}^-(A)$  that contains as subsets the largest number of prefixes  $A' \in \mathcal{P}$  not contained in any selected set so far. The algorithm terminates when all prefixes in  $\mathcal{P}$  have been covered. See Appendix B for details.

Combining these ideas, we obtain the improved Set Cover based Inclusion-Exclusion Algorithm, ALGIE (Algorithm 3). Recall that this algorithm is used in ALGDAG as a way to obtain improved edge probability estimates. Moreover, the number of samples required by ALGIE depends on the size of the set covers found in the algorithm; hence, if all set covers encountered when ALGIE is called by ALGDAG are small enough, the number of samples needed is small.

<sup>5</sup> Note, however, that some (or many) of the intersections may be empty, so the actual contributing number may be smaller.

**Algorithm 3:** The Set Cover based Inclusion-Exclusion Algorithm (ALGIE( $\widehat{G}_j, A, z; \epsilon, \delta$ ))**Input:** Graph  $\widehat{G}_j = (\widehat{V}_j, \widehat{E}_j)$ , node/set  $A$ , item  $z$ ,  $\epsilon$ , and  $\delta$ **Output:** An estimate of the edge probability  $\widehat{e}_{A \oplus z}$ 

/\* Find a minimum set cover \*/

1 Let  $\widehat{\mathcal{P}} = \{A' \in \widehat{V}_j \mid A' \subsetneq A, A' \cup \{z\} \in \widehat{V}_j\}$  and  $\mathcal{S}^-(A) = \{A' \subsetneq A \mid |A'| = |A| - 1\}$ .2 Find an approximately optimal set cover  $\widehat{\mathcal{C}} = \text{GREEDYMINCOVER}(\widehat{\mathcal{P}}, \mathcal{S}^-(A))$ . (See Appendix B.)

/\* Estimate the probabilities for all subsets in Inclusion-Exclusion \*/

3 Let  $m = \left\lceil \frac{2^{2|\widehat{\mathcal{C}}|-1} \cdot (\ln(1/\delta) + (|\widehat{\mathcal{C}}|+1) \cdot \ln(2))}{\epsilon^2} \right\rceil$ .4 Offer the assortment  $\mathcal{N} \setminus A$  to  $m$  consumers. Let  $\widehat{q}_z(\mathcal{N} \setminus A)$  be the fraction of consumers who choose  $z$  under assortment  $\mathcal{N} \setminus A$ .5 **for every**  $B \subseteq \widehat{\mathcal{C}}, B \neq \emptyset$  **do**6     Offer the assortment  $\mathcal{N} \setminus \bigcap_{B \in \mathcal{B}} B$  to  $m$  consumers.7     Let  $\widehat{q}_z(\mathcal{N} \setminus \bigcap_{B \in \mathcal{B}} B)$  be the fraction of consumers who choose  $z$  under assortment  $\mathcal{N} \setminus \bigcap_{B \in \mathcal{B}} B$ .8 Let  $\widehat{e}_{A \oplus z} = \widehat{q}_z(\mathcal{N} \setminus A) - \sum_{B \subseteq \widehat{\mathcal{C}}, B \neq \emptyset} (-1)^{|B|+1} \cdot \widehat{q}_z(\mathcal{N} \setminus \bigcap_{B \in \mathcal{B}} B)$ .9 **return** the edge probability  $\widehat{e}_{A \oplus z}$ .**7.3. Guarantees for ALGDAG**

The key guarantees for ALGDAG (Algorithm 2) are summarized by the following theorem, which we prove in Appendix D. In the statement of the theorem, recall that  $G_{n_0}^F$  denotes the subgraph of the true DAG induced by the first  $n_0$  layers and only the prefixes appearing for frequent types.

**THEOREM 3 (Performance Guarantee of ALGDAG).** *Let  $n_0 = \alpha n$  for some constant  $\alpha \in (0, 1)$ , and assume that the total probability of the rare rankings is at most  $\rho < \kappa/4$ . Under the  $(\kappa, \rho)$ -General Model, Algorithm 2 (run with parameters  $\epsilon > 0$  and  $\delta \in (0, 1)$ ) returns an estimated choice model  $(\widehat{G}_{n_0}, \widehat{p}, \widehat{e})$ , such that with probability at least  $1 - \delta$ , the following all hold:*

$$\widehat{V}_{n_0} = V_{n_0}^F \quad \widehat{E}_{n_0} = E_{n_0}^F \quad \max_{A \in V_{n_0}^F} |\widehat{p}(A) - p(A)| \leq \epsilon + n_0 \rho \quad \max_{A \in V_{n_0}^F, A \cup \{z\} \in V_{n_0}^F} |\widehat{e}_{A \oplus z} - e_{A \oplus z}| \leq \epsilon/n_0 + \rho.$$

Furthermore, under the  $(\kappa, \rho)$ -Random Model, writing  $\nu = \log_\alpha(1/2) > 0$ , with probability at least  $1 - 2\delta$ , the total number of queries and the total computation time are bounded by

$$O\left(\frac{K^{1+4\nu} \cdot n^{4+2\nu} \cdot \log(nK/\delta)}{\delta^{2\nu} \cdot \min(\epsilon, \kappa)^2}\right).$$

ALGDAG works for both the General and Random Models. However, the size of all set covers is *guaranteed* to be logarithmically small only for the Random Model (with high probability) — this results in a polynomial total number of samples in that case. Nevertheless, even for the General Model, if the size of all set covers is small, the required number of samples is also small. Furthermore, notice that in ALGIE, after finding the approximate minimum set cover, the algorithm knows the size of the set cover. Therefore, ALGIE (and consequently also ALGDAG) can easily diagnose when a larger number of samples is needed to obtain good accuracy; alternatively, the algorithm can use a theoretically insufficient number of samples and return the result with some confidence interval around the probability of each node and edge.

## 8. Implications of Approximations in DAG Representation

The DAG representation serves several important purposes. Firstly, it can act as a potentially valuable visualization of consumer preferences. Secondly, it has a practical application: using the DAG representation, one can readily identify all top- $j$  sets of frequent types for a given  $j$ , represented as nodes at level  $j$ . These sets can then be used to create an assortment containing at least one top- $j$  item for each frequent type, or similar assortment design tasks.

Our findings in Section 6 reveal an equivalence between the DAG representation and choice probabilities. Thus, an exact DAG facilitates the precise calculation of choice probabilities, and vice versa. This in turn enables the application of the DAG framework in optimization problems like assortment planning under various constraints including *user and item fairness*. Such constraints ensure that in expectation, the offered set maintains a sufficiently high market share (probability of purchase) for each frequent user type and for each item. See Chen et al. (2022, 2023), Lu et al. (2023) for recent works that study such constraints.

While an exact DAG representation enables the computation of exact choice probabilities, ALGDAG computes an approximate DAG, raising the question of how this approximation impacts the computation of choice probabilities. Observe that there are three sources of approximation as characterized in Theorem 3:

1. Only prefixes corresponding to frequent types will occur in the representation output by ALGDAG.
2. The vertex and edge probabilities are estimated only up to some errors.
3. The DAG is truncated at level  $n_0 = \alpha n$ .

The first error source is most benign. Rare types contribute a very small probability (at most  $\rho$  total), so not including their prefixes in calculations can change the choice probability estimates by at most  $\rho$ .

The impact of the second source can also be bounded. In Proposition 2, each frequent type  $\pi$  can contribute at most one term to the sum, because  $\pi$  uniquely defines the set  $A$  such that  $\pi \in \Theta_{A \oplus z}$ , as the set of all items preceding  $z$  in the ranking  $\pi$ . As a result, the sum can have at most  $K$  non-zero terms, and each is estimated, by Theorem 3, to within an additive error at most  $\epsilon/n_0 + \rho$ . Thus, the total estimation error due to errors in estimates of  $e_{A \oplus z}$  can be at most  $K \cdot (\epsilon/n_0 + \rho)$ . The first term can be made small by decreasing  $\epsilon$ ; for the second term, one needs an assumption that  $K\rho$  is not too large, i.e., the rare types make up only a small fraction of the population in total.

Most problematic is the truncation at level  $n_0 = \alpha n$ . Indeed, if there is a (very) frequent type  $\pi$ , and the assortment offered comprises the two least favorite items of  $\pi$ , the available information will not allow for any accurate estimation of the probability with which this type — and by extension the population — chooses one item vs. the other. More generally, any assortment  $S$  such that all items in  $S$  are among the bottom  $1 - \alpha$  fraction of types whose total probability mass is large enough will not allow an estimation of the probability with which any particular item  $z \in S$  is chosen. However, the converse is also true: if  $S$  *does* contain, for each frequent type  $\pi$ , at least one item ranked in the top  $\alpha$  fraction of items, then the truncation

is not problematic from the perspective of computing  $q_z(S)$ . To see this, recall that by Proposition 2, we have  $q_z(S) = \sum_{A: A \cap S = \emptyset} p(\Theta_{A \oplus z})$ . Now consider any set  $A$  of size more than  $n_0$  with  $A \cap S = \emptyset$ , whose probability thus was not estimated. Any type  $\pi \in \Theta_{A \oplus z}$  must have all of  $A$  preceding all of  $\mathcal{N} \setminus A$ , and in particular all of  $S$ . Because  $|A| > n_0$ , this means that no items of  $S$  appears in the first  $n_0$  items of  $\pi$ , so by the assumption,  $\pi$  cannot be a frequent type. Therefore,  $\Theta_{A \oplus z}$  cannot contain any frequent types for any  $A$  that was not estimated, and the estimate of  $q_z(S)$  will be accurate up to rare types.

From a practical perspective, assortments under which a frequent type dislikes all items (in the sense of ranking all of them in the bottom  $1 - \alpha$  fraction of the rankings) are perhaps of less interest to evaluate. In particular, this applies if consumers would prefer not to purchase any items in such cases. In fact, if we assume that a no-purchase option is always included as an “item”, and consumers dislike their bottom  $1 - \alpha$  fraction of items enough to always prefer the no-purchase option, then the no-purchase option will be an element of every set  $S$  which is ranked in the top  $n_0$  items of each type  $\pi$ . As a result, the discussion from the previous paragraph exactly applies.

Under the random model, choice sets are also unlikely to face this issue. For a choice set of size  $S$ , the probability that a random type ranks all of  $S$  in the bottom  $1 - \alpha$  fraction of items is at most  $(1 - \alpha)^{|S|}$ . By a union bound over all frequent types, the probability that at least one frequent type is affected is at most  $K(1 - \alpha)^{|S|}$ . Whenever the choice set  $S$  is sufficiently large, this quantity will be small enough that most choice probabilities will be estimated accurately.

Thus, under various natural modeling assumptions, the impact of all types of errors is limited. Nonetheless, we believe that eliminating the third source of error — i.e., trying to not truncate the DAG at all — is perhaps the most interesting technical direction for future work.

## 9. Experiments

In this section, we describe two experiments. The first experiment complements the theoretical discussion in Section 8 and focuses the impact of approximation in the DAG representation on the estimated choice probabilities. The second experiment focuses on the accuracy of ALGDAG in estimating the true DAG. Our goal in the second experiment is to investigate the benefit of actively learning the choice model, compared to learning the best choice model given the available transaction data, as previously done in Farias et al. (2013), van Ryzin and Vulcano (2015), and van Ryzin and Vulcano (2017).

### 9.1. Impact of Approximation on Estimated Choice Probabilities

**Setup.** We generate choice models as follows. We consider 5 frequent types and  $n \in \{8, 16\}$  items. The total probability  $\rho$  of noise is chosen from  $\{0.001, 0.01, 0.05\}$ . For any  $\rho$ , there are 20 noisy types.<sup>6</sup> For each of the frequent and noisy types, a ranking over the  $n$  items is chosen i.i.d. uniformly at random. The probability

<sup>6</sup> Separate experiments have shown that changing the number of rare types does not impact our results given a fixed  $\rho$ .

of the frequent types is drawn from a symmetric Dirichlet distribution whose coefficient of variation is 0.1, multiplied by  $(1 - \rho)$ . The probability of the noisy types is distributed as a symmetric Dirichlet distribution whose coefficient of variation is 0.1, multiplied by  $\rho$ . For each of  $\rho \in \{0.001, 0.01, 0.05\}$ , 100 instances of the described choice models are generated.

**Metric.** For each choice model, we uniformly randomly generate 100 sets of size  $k = 4$ . (The results are similar when  $k$  is set to 3 or 5.) We then calculate the average L1 error in the estimated choice probabilities for the randomly chosen set. More specifically, let  $S$  be the (randomly chosen) set and  $(\hat{q}_z(S))_{z \in S}$  be the estimated choice probabilities for the elements in this set. The L1 error is  $\sum_{z \in S} |\hat{q}_z(S) - q_z(S)|$ . For each choice model, we compute the average L1 errors, taken over the 100 draws of random sets.

**Table 1** L1 error of choice probabilities for  $n = 8$  and random sets of size 4 with  $\kappa = 0.01$ , and  $\delta = 0.05$ .

$n_0$	$\rho$	$\epsilon$	Statistics				
			Mean	Std	25%	50%	75%
3	0.001	0.001	0.071	0.011	0.064	0.072	0.079
		0.01	0.072	0.010	0.065	0.072	0.080
	0.01	0.001	0.077	0.012	0.070	0.078	0.084
		0.01	0.076	0.010	0.070	0.077	0.082
	0.05	0.001	0.087	0.012	0.080	0.086	0.095
		0.01	0.085	0.012	0.077	0.085	0.092
4	0.001	0.001	0.014	0.005	0.010	0.014	0.017
		0.01	0.014	0.005	0.010	0.014	0.018
	0.01	0.001	0.020	0.006	0.015	0.020	0.023
		0.01	0.020	0.005	0.016	0.019	0.022
	0.05	0.001	0.031	0.006	0.027	0.031	0.035
		0.01	0.032	0.005	0.028	0.031	0.036
5	0.001	0.001	0.001	0.000	0.001	0.001	0.001
		0.01	0.001	0.000	0.001	0.001	0.001
	0.01	0.001	0.006	0.001	0.005	0.005	0.006
		0.01	0.006	0.001	0.005	0.006	0.006
	0.05	0.001	0.018	0.003	0.016	0.017	0.019
		0.01	0.017	0.002	0.016	0.017	0.019

**Table 2** L1 error of choice model for  $n = 16$  and random sets of size 4 with  $\kappa = 0.01$ , and  $\delta = 0.05$ .

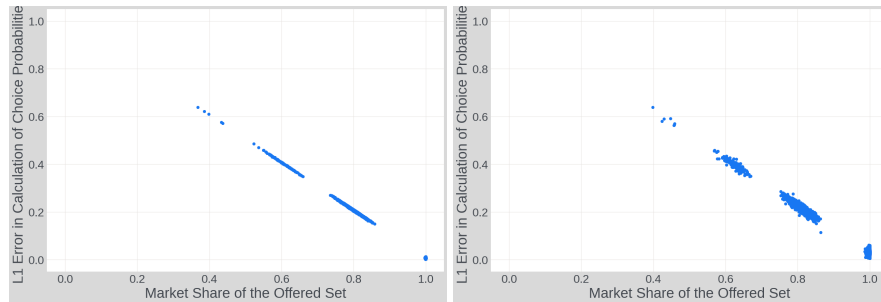
$n_0$	$\rho$	$\epsilon$	Statistics				
			Mean	Std	25%	50%	75%
6	0.001	0.001	0.118	0.014	0.108	0.117	0.126
		0.01	0.118	0.015	0.106	0.118	0.127
	0.01	0.001	0.121	0.014	0.110	0.121	0.130
		0.01	0.122	0.015	0.111	0.124	0.134
	0.05	0.001	0.142	0.012	0.133	0.141	0.150
		0.01	0.141	0.013	0.132	0.141	0.150
8	0.001	0.001	0.040	0.009	0.034	0.039	0.045
		0.01	0.039	0.008	0.033	0.039	0.045
	0.01	0.001	0.046	0.008	0.040	0.045	0.050
		0.01	0.048	0.008	0.042	0.048	0.051
	0.05	0.001	0.066	0.010	0.059	0.065	0.072
		0.01	0.066	0.008	0.060	0.067	0.072
10	0.001	0.001	0.009	0.004	0.007	0.009	0.011
		0.01	0.009	0.004	0.007	0.009	0.012
	0.01	0.001	0.016	0.004	0.014	0.016	0.018
		0.01	0.016	0.005	0.013	0.016	0.019
	0.05	0.001	0.038	0.005	0.035	0.038	0.041
		0.01	0.037	0.005	0.034	0.037	0.041

**9.1.1. Results** Tables 1 and 2 present the average (mean) of the L1 errors, along with their standard deviation (std), and 25th, 50th, and 75th percentiles for  $n = 8$  and  $n = 16$ , respectively. This analysis is conducted for values of  $\rho \in \{0.001, 0.01, 0.05\}$ ,  $\epsilon \in \{0.001, 0.01\}$ ,  $\kappa = 0.01$ , and  $\delta = 0.05$ . In Table 1, for certain values of  $n_0$ , the results for  $\rho = 0.001$  are omitted, because all numbers are zero after rounding them to three decimal places. In Table 1, corresponding to  $n = 8$ , we consider  $n_0$  values from the set  $\{3, 4, 5\}$ . These values are then doubled in Table 2 when the number of items is  $n = 16$ . We notice relatively small L1 errors of choice probabilities. For example, even when  $n = 8$  and  $n_0 = 3$  with a noise level of 5%, the mean L1 error for a random set of size 4 is only 0.031, with 75% of the samples having an L1 error of less than 0.035. Further observations include:

- **Impact of  $\rho$  and  $\epsilon$ :** As expected, the mean and standard deviation increase in  $\rho$  and  $\epsilon$ , as there is more noise in the model or fewer samples are drawn. However, this increase is only small.
- **Impact of truncation:** As the truncation level  $n_0$  increases (so the DAG representation has more levels), there is a general trend of decreasing mean L1 error, standard deviation, and percentiles. Consistent with the discussion in Section 8, the impact of  $n_0$  is significantly larger than that of  $\rho$  and  $\epsilon$ .
- **Impact of  $n$ :** Comparing Tables 1 and 2, doubling the number of items from  $n = 8$  to  $n = 16$  leads to higher mean errors and increased variability in the errors. In the limit of large  $n$ , the ratio  $n_0/n$  is the primary indicator of errors, because it captures the probability that a random element of the set is in the DAG. However, for small  $n$ , because the set  $S$  contains distinct elements, the probability that none of them is in the DAG is significantly smaller than the estimate  $(1 - n_0/n)^k$ , implying a higher market share (see next paragraph) and thus lower error.

Tables 1 and 2 show that truncation is the primary source of L1 errors. To shed light on the impact of truncation, in Figure 4, we present the L1 error of choice probabilities versus the *market share* of a set of size 4 when  $n = 16$ ,  $n_0 = 8$ , and  $\rho \in \{0.01, 0.05\}$ . Here, the market share of a set  $S$  given the truncation level  $n_0$  is equal to the probability  $\sum_{\pi: \{\pi(1), \dots, \pi(n_0)\} \cap S \neq \emptyset} p(\pi)$  that at least one of the items in  $S$  is in the first  $n_0$  layers of the DAG. The market share of  $S$  in this sense can be considered as a measure of the extent to which the assortment provides at least one attractive item (within the top  $n_0$  levels) to customers.

Figure 4 shows that the larger the market share, the smaller the L1 error. In fact, the relationship between the L1 error and the market share of a set is linear. The same observation is also confirmed for every combination of  $n$  and  $n_0$  presented in Tables 1 and 2. The impact of  $\rho$ , which can be viewed as the observed deviation from the linear model, is quite negligible; this is consistent with our findings in Tables 1 and 2. Similar observation hold for the dependence on  $\epsilon$  (not shown here).



**Figure 4** L1 error in choice probabilities vs. market share of the set. In both figures,  $n = 16$ ,  $n_0 = 8$ ,  $\epsilon = 0.01$ , and  $\kappa = 0.01$ . The only difference is that in the left figure,  $\rho = 0.01$ , while in the right figure,  $\rho = 0.05$ .



## 9.2. The Value of Active Learning

In our second set of experiments, we evaluate the performance of ALGDAG empirically on both synthetic and real-world datasets, shedding light on the value of active learning. Our experiments show that, in addition to being easy to implement, ALGDAG can recover the true DAG accurately (with low estimation errors) with high probability using a reasonable number of samples. They further show that actively learning the choice model gives better performance empirically than learning the model directly from pre-determined/given transaction data.

**Synthetic setup.** For the synthetic analyses, we use a setup similar to the first experiment, with a few minor differences. Due to computational challenges posed by the benchmark algorithm (defined shortly), we only conduct our analysis for  $n = 8$ . (In Appendix G, we further evaluate ALGDAG for a larger number of types and items, to observe how ALGDAG performs when the instance size scales up.) Similar to the first experiment, the total probability  $\rho$  of noise is chosen from  $\{0, 0.001, 0.01, 0.05\}$ ; note that we add the noiseless condition  $\rho = 0$  here.

**Real-world setup.** We further evaluate the performance of ALGDAG on the sushi data set (Kamishima et al. (2005)). This publicly available data set describes preferences over 10 popular types of sushi based on a consumer survey that involved 5000 people specifying their exact ranking. In the sushi data set, unlike the data one would acquire in an active learning manner, a user declares his/her entire ranking, whereas from transaction data, we only get to see the most popular sushi among those offered.

In the sushi data set, there are 4926 distinct rankings. Taking  $\kappa = 0.0001$ , there are 90, 561, 1800, and 3273 frequent prefixes for the top  $n_0 = 2, 3, 4$  and 5 items, respectively, all with zero noise probability, i.e.  $\rho = 0$ .<sup>7</sup> Our choice of  $n_0$  is mainly due to our benchmark algorithm (we elaborate on this more in the next section), which is computationally inefficient when estimating more than the top 5 items in our experiments. Observe that in this setup, there are a lot more distinct types and prefixes compared to the synthetic setting. This causes a lot more interference, which makes it harder for ALGDAG to estimate each node properly. Furthermore, the set of rankings is obviously not generated i.i.d. uniformly at random. We use the empirical frequency of each prefix in the data set as our ground truth, so for each  $n_0 \in \{2, 3, 4, 5\}$  that we consider, we get the set of types of size  $n_0$  with its corresponding probability distribution.

**Metrics.** We compare the true DAG representation of the choice model with the DAG produced by ALGDAG for various values of  $n_0$  (the number of top positions to estimate). We consider two metrics to measure the performance of ALGDAG. The first one is the number of nodes in the DAG that are added wrongly (false positive) or not inferred (false negative). The second one is the maximum probability discrepancy/difference between the true DAG and the inferred DAG.

<sup>7</sup> Increasing the threshold  $\kappa$  for frequent types causes the total probability of noise to be much larger than  $\kappa$ ; hence we choose  $\kappa = 0.0001$  to be our threshold. For instance, when  $\kappa = 0.0002$ , we have  $\rho = 0.4754$ , which is more than 2000 times the threshold for frequent types.

**9.2.1. Benchmark** As our benchmark, we implemented the primal linear program (LP) from Farias et al. (2013) to find the set of types. Farias et al. (2013) summarize the given transaction data in an  $m$ -dimensional vector  $\mathbf{y} = A\boldsymbol{\lambda}$ , where  $A \in \{0, 1\}^{m \times n!}$  represents the relationship between the observed data ( $\mathbf{y}$ ) and the choice model.<sup>8</sup> Here, the choice model is represented by the vector  $\boldsymbol{\lambda}$ , which is the variable representing the probability distribution over all possible types of preferences. Unlike ALGDAG, which builds the DAG level by level using active learning, this method determines the distribution over the whole ranking (containing  $n$  items) using an offline data set. That is, in our synthetic setting, for a given offline data set, it returns a distribution over all possible  $8! = 40320$  candidate rankings. In the sushi setting, since we only run the benchmark algorithm to estimate the top 5 items, we limit our type space to the set of all possible rankings of five items; in this case, the benchmark returns a distribution over all  $5! \cdot \binom{11}{5} = 55440$  possible top 5 items. Thus, in the sushi data set, we use a smaller search space for the types instead of searching over all  $11! = 39,916,800$  rankings since ALGDAG also only estimates up to the top 5 items. This is beneficial for the calculation complexity for the benchmark since it reduces the number of decision variables in the LP.<sup>9</sup> See more details about the benchmark implementation in Section G.

To evaluate the benchmark, we generated two types of offline transaction datasets for the synthetic setting. The first dataset aims to replicate common stockout scenarios observed in the real world, considering the impact of stockouts on demand and revenue (Caine and Plaut 1976, Fitzsimons 2000, Kim and Lennon 2011). Here, we assume that 3 out of  $n$  items are unavailable, leaving consumers with  $(n - 3)$  items to choose from. This results in  $\binom{n}{3}$  distinct assortments, with each assortment shown to  $m$  consumers. With 8 items, this yields 56 distinct assortments. The second dataset comprises pairwise comparison data representing the fraction of consumers who prefer one item over another, as seen in previous research (Farias et al. 2013). This dataset contains  $\binom{n}{2}$  distinct assortments, resulting in 28 distinct assortments when there are 8 items.<sup>10</sup>

In the sushi setting, we generated two types of offline transaction data: assortments with 3 stockouts (resulting in 8 items) and pairwise assortments. However, when using pairwise assortments, the linear programs often remained unsolved due to an undetermined linear system, especially with the large number of distinct rankings in the sushi dataset (4926). Therefore, we focus solely on the results from the offline assortments with 3 stockouts for the sushi dataset to provide additional advantage to the offline benchmark.

<sup>8</sup> Specifically, the  $i^{\text{th}}$  element of  $\mathbf{y}$ , which is associated with a set  $S$  and an item  $z$ , represents the fraction of consumers who choose an item  $z$  given assortment  $S$ . Then, the element in the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column of matrix  $A$ , where the  $i^{\text{th}}$  row is associated with a set  $S$  and an item  $z$ , is 1 if consumers with type  $j$  prefer  $z$  among all items in  $S$ , and 0 otherwise.

<sup>9</sup> In the synthetic setting, the problem is much easier because the number of types is at most 35, while in the sushi setting, there are at least 3273 distinct types when considering the top 5 items. Thus, in the synthetic data set, unlike the sushi data set, we consider all  $n!$  rankings in the benchmark LP.

<sup>10</sup> When applying ALGDAG to the synthetic dataset, we find that it uses a maximum of 20 distinct assortments, even with  $n_0 = 5$ . This suggests that ALGDAG employs fewer distinct queries compared to offline datasets. However, through active learning, as we will show later, ALGDAG significantly outperforms the benchmark.

Nevertheless, as we will demonstrate later, even in the synthetic dataset, the benchmark performed worse on the pairwise dataset compared to the 3-stockout dataset.

**9.2.2. ALGDAG Implementation** In the synthetic setting, we want to estimate the top  $n_0 = 3, 5$  positions when the number of items is 8, while in the sushi setting, we want to estimate the top  $n_0 = 2, 3, 4, 5$  positions. In the synthetic data set, we consider as frequent the types whose probability is at least  $\kappa = 0.01$ ; this is a consequence of our choice of Dirichlet parameters for the probability distribution of the frequent types in the synthetic setting.<sup>11</sup> Meanwhile, in the sushi setting, we choose  $\kappa = 0.0001$  by looking at the smallest type probability in the ranking. Note that in our experiment, we do not always have  $\rho < \kappa/4$  for our chosen  $\rho \in \{0, 0.001, 0.01, 0.05\}$ . While this assumption is required for our theoretical guarantee in Theorem 3, by violating it, we can observe how robust ALGDAG is.

In simulating active learning, given a set of (true) synthetic parameters, one consumer is drawn in each round, and the item she chooses based on her preference is observed. The algorithm gets to decide which assortment is offered for each consumer. To run ALGIE, we set the number of samples for each ALGIE run to be at most  $10k$ . Such a choice would likely correspond to how a practitioner would use ALGIE, and lets us observe the impact of a number of samples that is significantly smaller than that required for theoretical guarantees. The experiment is run 100 times for each value of  $n_0$ .

Noise	ALGDAG, $m \leq 10k$		Farias et al. (2013), 3 stockout		Farias et al. (2013), pairwise data	
	$n_0 = 3$	$n_0 = 5$	$n_0 = 3$	$n_0 = 5$	$n_0 = 3$	$n_0 = 5$
0	$0.0083 \pm 0.00040$	$0.0092 \pm 0.00041$	$0.2866 \pm 0.01100$	$0.2988 \pm 0.01214$	$0.3785 \pm 0.00916$	$0.3884 \pm 0.01032$
0.001	$0.0099 \pm 0.00033$	$0.0111 \pm 0.00032$	$0.3054 \pm 0.01110$	$0.3359 \pm 0.01183$	$0.3953 \pm 0.01132$	$0.4026 \pm 0.01201$
0.01	$0.0103 \pm 0.00036$	$0.0113 \pm 0.00038$	$0.3100 \pm 0.01134$	$0.3585 \pm 0.01423$	$0.4280 \pm 0.01141$	$0.4186 \pm 0.01160$
0.05	$0.0105 \pm 0.00039$	$0.0114 \pm 0.00039$	$0.3320 \pm 0.01321$	$0.3878 \pm 0.01477$	$0.4058 \pm 0.01078$	$0.4414 \pm 0.01178$

**Table 3** Maximum probability discrepancy, over all nodes, between the true DAG and the DAG returned by ALGDAG or the benchmark, when there are 5 frequent types and 8 items, averaged over 100 instances. Here,  $m = 10k$ .

### 9.3. Results

**9.3.1. Synthetic Data Set.** We present our results in Tables 3 and 4. In Table 3, we look at the maximum probability discrepancy out of all nodes between the DAG returned by the algorithm and the true DAG. ALGDAG outperforms the benchmark for both offline data sources: the 3 stockout and pairwise data. In general, the results for both algorithms (ALGDAG and the benchmark) get worse when the noise probability increases. Further, in Table 3, we see that the maximum probability discrepancy for the DAG returned by ALGDAG, which is attained when the noise probability is the largest, i.e., 0.05, is at most 0.00892 on

<sup>11</sup> Recall that in the synthetic setting, the probability of the frequent types is drawn from a symmetric Dirichlet distribution whose coefficient of variation is 0.1, multiplied by  $(1 - \rho)$ . The implication is that most of the probability of a frequent type generated is bigger than 0.01, so we can set  $\kappa = 0.01$ .

average (when  $n_0 = 5$ ). Moreover, the maximum noise discrepancy is larger for  $n_0 = 5$  than for  $n_0 = 3$ . This shows that as the number of top items being inferred increases, the problem becomes harder. This is similar to the behavior predicted by Theorem 3, where more samples are needed to obtain the same level of accuracy. Comparing the benchmarks utilizing two distinct data sources — specifically, the 3 stockout data set and the pairwise data set — we observe that the benchmarks exhibit superior performance when employing the 3 stockout offline data set as opposed to the pairwise data set. The former has more variation in the transaction data with 56 distinct assortments, therefore containing more information.

Noise	ALGDAG, $m \leq 10k$		Farias et al. (2013), 3 stockout		Farias et al. (2013), pairwise data	
	$n_0 = 3$	$n_0 = 5$	$n_0 = 3$	$n_0 = 5$	$n_0 = 3$	$n_0 = 5$
0	1.58% $\pm$ 0.149%	1.73% $\pm$ 0.209%	30.73% $\pm$ 0.315%	35.30% $\pm$ 0.633%	36.71% $\pm$ 0.964%	41.90% $\pm$ 0.796%
0.001	4.67% $\pm$ 0.154%	5.04% $\pm$ 0.228%	31.51% $\pm$ 0.333%	41.01% $\pm$ 0.744%	37.19% $\pm$ 0.991%	45.25% $\pm$ 0.972%
0.01	5.40% $\pm$ 0.167%	5.68% $\pm$ 0.234%	37.67% $\pm$ 0.366%	41.27% $\pm$ 0.656%	44.61% $\pm$ 1.278%	46.48% $\pm$ 0.994%
0.05	5.73% $\pm$ 0.207%	6.04% $\pm$ 0.252%	38.12% $\pm$ 0.363%	49.33% $\pm$ 0.770%	46.99% $\pm$ 1.417%	52.51% $\pm$ 1.103%

**Table 4** Percentage of vertices that are different between the true DAG and the DAG returned by ALGDAG or the benchmark, averaged over 100 instances. Here,  $m = 10k$ .

In Table 4, we look at the percentage of vertices that are added wrongly or not inferred, i.e., the number of vertices that appear only once (either in the true DAG or in the DAG returned by ALGDAG, but not in both) on level  $n_0$ , as a percentage of the number of vertices on level  $n_0$  in the true DAG formed by the frequent types. Compared with the benchmark, ALGDAG produces fewer vertices that are being added wrongly or not inferred compared to the benchmark, with both data sources.<sup>12</sup> For example, when  $n_0 = 5$  and the noise level is 0.001, while the percentage of different vertices is 5.04% in ALGDAG, it is 41.01% for the benchmark with 3 stockout data and 45.25% for the benchmark with pairwise data. Further, as before, the benchmark with 3 stockout data outperforms the one with pairwise data due to the existence of more variation in the data. As expected, for both ALGDAG and the benchmark, the percentage of different vertices gets bigger as the total noise probability increases.

	ALGDAG, $m \leq 10k$				Farias et al. (2013), 3 stockout			
	$n_0 = 2$	$n_0 = 3$	$n_0 = 4$	$n_0 = 5$	$n_0 = 2$	$n_0 = 3$	$n_0 = 4$	$n_0 = 5$
max. prob. differences	0.0110 $\pm$ 0.00023	0.0258 $\pm$ 0.00054	0.0411 $\pm$ 0.00073	0.0765 $\pm$ 0.00134	0.0203 $\pm$ 0.00055	0.0320 $\pm$ 0.00090	0.0472 $\pm$ 0.00105	0.0950 $\pm$ 0.00154
% of different vertices	1.67% $\pm$ 0.174%	8.88% $\pm$ 0.209%	9.45% $\pm$ 0.124%	9.58% $\pm$ 0.169%	23.36% $\pm$ 0.614%	56.77% $\pm$ 0.535%	74.46% $\pm$ 0.435%	86.16% $\pm$ 0.476%

**Table 5** Maximum probability discrepancy over all nodes and percentage of incorrect vertices, between the true DAG and the DAG returned by ALGDAG or the benchmark, averaged over 100 instances run on the sushi data.

<sup>12</sup> Additional experiments — not discussed here — show that the number of different vertices goes down to 0 when we increase  $m$  to 50k.

**9.3.2. Sushi Data Set.** Table 5 summarizes our results on the sushi data set, showing the maximum probability differences and the percentages of different vertices between the DAG returned by ALGDAG and the true DAG, averaged over 100 runs. ALGDAG far outperforms the benchmark in both metrics: it returns node probabilities within 0.0765 of the actual probability on average when we look at the fifth level (i.e.,  $n_0 = 5$ ), compared to 0.0950 for the benchmark. Similarly, when we look at the number of vertices that are either added wrongly or not inferred, on average, the percentage for ALGDAG is nine times smaller than for the benchmark, for  $n_0 = 5$ . These results are surprising since we are only using  $m = 10k$ ; they indicate that even with such a small number of samples (far smaller than what is needed in the theoretical guarantee), ALGDAG can estimate the true distribution of types with good accuracy in the sushi data. Note that this holds even though the types are not drawn i.i.d. uniformly at random for the sushi data.

**9.3.3. Concluding Remarks for the Second Set of Experiments.** Overall, we observe that the performance of both algorithms, ALGDAG and the offline benchmark, gets worse as the total probability of noise (i.e.,  $\rho$ ) gets larger or the number of items being estimated (i.e.,  $n_0$ ) gets bigger. Furthermore, the experiments show that ALGDAG can get a small probability discrepancy and few “wrong” vertices even with a small number of samples ( $m = 10k$ ). When the number of types increases, the performance gets worse but is still comparable, which can be seen in the results for the sushi data set and the bigger synthetic data set with 15 frequent types and 20 items (Appendix G). ALGDAG also performs well when the number of items is larger than the number of types, as is the case, for example, in the sushi data; this is true as long as there are not too many types with similar preferences causing a lot of interference when ALGDAG is run. In all scenarios, ALGDAG outperforms the benchmark despite using a small number of distinct queries.

## 10. Conclusion, Discussions, and Future Directions

Non-parametric choice models, while able to capture general choice models, are difficult to estimate using only offline transaction data. This is primarily due to the lack of variation in the data. In many settings, especially on online platforms, one can influence the process of acquiring data based on past observations; this is also known as an active learning process. Motivated by this, we studied the problem of estimating a non-parametric choice model using active learning. In addition to product ranking, non-parametric choice models have also been used in other applications, such as a non-parametric joint assortment and price choice model with consideration set (Jagabathula and Rusmevichientong (2017)) and non-parametric demand for dynamic pricing (Perakis and Singhvi (2019)). In future work, it would be interesting to see how an active learning approach applies to those problems.

We present a negative result showing that such non-parametric choice models might be indistinguishable regardless of the sequence of assortments offered in the active learning process. To overcome this obstacle, we present a directed acyclic graph (DAG) representation of the choice model which captures as much as can be uniquely identified about the distribution when the types of consumers are unobservable and each

consumer only makes one choice. We show that the (exact) DAG representation can be used to compute the (exact) choice probabilities and vice versa, implying that this representation encapsulates all the information regarding the choice model that can be deduced from the available data.

We then design an efficient algorithm to accurately estimate the DAG representation of non-parametric choice models using a polynomial-sized data set. The algorithm learns the distribution over the most popular items and their corresponding probabilities by actively choosing assortments to offer the next arriving consumer based on past observations. The algorithm employs an inclusion-exclusion method to better estimate each ranking probability. We also show that ALGDAG outperforms a non-active learning algorithm on synthetic and real-world (sushi) data sets.

Our work raises several immediate questions for technical follow-up work. First, the factor  $n_0$  in the error bound of Theorem 3 seems like it could perhaps be avoided. Notice, however, that rare types could indeed impact the estimate of  $e_{A \oplus z}$  for multiple  $A$ , for instance those rare types that rank  $z$  first (or early). Similarly, while for many practical applications, it may suffice to learn the first  $n_0 = \alpha n$  items of each ranking (see the discussion in Section 8), both from a theory and practice perspective, it would be interesting to understand whether learning the entire rankings is possible. To learn about the last few items in rankings, it is necessary to offer small sets to consumers, but if types are sufficiently frequent, it may be possible to obtain data on those items. However, the approach based on Set Cover is unlikely to succeed: when  $n_0$  is large, it is indeed likely that the required set covers are large, and hence, the Inclusion-Exclusion formula requires exponentially many terms, and thus exponentially small error in each term to compensate.

Our polynomially small bounds on the required samples were only guaranteed to apply under the  $(\kappa, \rho)$ -Random Model. It would be interesting to investigate whether small sample bounds can be obtained under a fully adversarial model. Some preliminary work suggests that this may be difficult. Yet, although there is no polynomial sample bound for the general model, AlgDAG still requires a small number of samples as long as the approximate solution to the Set Cover instance is small.

Experimentally, it would be desirable to evaluate the performance of our algorithms, as well as alternatives, on larger and richer data sets. Ultimately, the goal should be to deploy them in a real-world setting and have them interact with actual consumers. In such a setting, there may be other considerations, such as the cost of exploration (i.e., offering consumers sets of items that are ranked low by many types, to disambiguate the rankings). Many other practical concerns such as inventory constraints are likely to arise.

Fundamentally, we view our work as a first step towards a much more comprehensive evaluation of active learning in the context of choice models and related settings. Our work opens up new avenues for future work, and we hope that it will lead to studying the framework in other Operations Research problems, such as product ranking and assortment planning.

## Acknowledgments

N.G. was supported in part by the Young Investigator Program (YIP) Award from the Office of Naval Research (ONR) N00014-21-1-2776 and the MIT Research Support Award. D.K. was supported in part by ARO MURI grant W911NF1810208. We thank Antoine Desir and Ningyuan Chen for their help and insightful comments.

## References

- Charu C. Aggarwal, Xiangnan Kong, Quanquan Gu, Jiawei Han, and Philip S. Yu. *Active learning: A survey*, pages 571–605. CRC Press, January 2014. ISBN 9781466586741. doi: 10.1201/b17320.
- Gagan Aggarwal, Jon Feldman, S. Muthu Muthukrishnan, and Martin Pál. Sponsored search auctions with markovian users. In *Proc. 4th Workshop on Internet and Network Economics (WINE)*, pages 621–628, 2008.
- Shipra Agrawal, Vashist Avadhanula, Vineet Goyal, and Assaf Zeevi. Mnl-bandit: A dynamic learning approach to assortment selection. *Operations Research*, 67(5):1453–1485, 2019.
- Dana Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.
- Ali Aouad, Vivek Farias, Retsef Levi, and Danny Segev. The approximability of assortment optimization under ranking preferences. *Operations Research*, 66(6):1661–1669, 2018.
- Ali Aouad, Retsef Levi, and Danny Segev. Approximation algorithms for dynamic assortment optimization models. *Mathematics of Operations Research*, 44(2):487–511, 2019.
- Yossi Aviv and Amit Pazgal. Pricing of short life-cycle products through active learning. *Under revision for Management Science*, pages 1–32, 2002.
- Moshe E. Ben-Akiva, Steven R. Lerman, et al. *Discrete choice analysis: theory and application to travel demand*, volume 9. MIT press, 1985.
- Jose Blanchet, Guillermo Gallego, and Vineet Goyal. A markov chain approximation to choice modeling. *Operations Research*, 64(4):886–905, 2016.
- G. J. Caine and Raymond H. Plaut. Optimal inventory policy when stockouts alter demand. *Naval Research Logistics Quarterly*, 23(1):1–13, 1976.
- Qinyi Chen, Negin Golrezaei, and Fransisca Susan. Fair assortment planning. *arXiv preprint arXiv:2208.07341*, 2022.
- Qinyi Chen, Jason Cheuk Nam Liang, Negin Golrezaei, and Djallel Bouneffouf. Interpolating item and user fairness in recommendation systems. *arXiv preprint arXiv:2306.10050*, 2023.
- Xi Chen, Chao Shi, Yining Wang, and Yuan Zhou. Dynamic assortment planning under nested logit models. *Production and Operations Management*, 30(1):85–102, 2021.
- Alexander Chernev. Decision focus and consumer choice among assortments. *Journal of Consumer Research*, 33(1): 50–59, 2006.
- David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.
- James M. Davis, Guillermo Gallego, and Huseyin Topaloglu. Assortment optimization under variants of the nested logit model. *Operations Research*, 62(2):250–273, 2014.

- Mahsa Derakhshan, Negin Golrezaei, Vahideh Manshadi, and Vahab Mirrokni. Product ranking on online platforms. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 459–459, 2020.
- Vivek F. Farias, Srikanth Jagabathula, and Devavrat Shah. A nonparametric approach to modeling choice with limited data. *Management science*, 59(2):305–322, 2013.
- Uriel Feige. A threshold of  $\ln n$  for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.
- Jacob Feldman, Alice Paul, and Huseyin Topaloglu. Assortment optimization with small consideration sets. *Operations Research*, 67(5):1283–1299, 2019.
- Jacob B Feldman and Huseyin Topaloglu. Revenue management under the markov chain choice model. *Operations Research*, 65(5):1322–1342, 2017.
- Gavan J. Fitzsimons. Consumer response to stockouts. *Journal of consumer research*, 27(2):249–266, 2000.
- Guillermo Gallego and Wentao Lu. An optimal greedy heuristic with minimal learning regret for the markov chain choice model. *Available at SSRN 3810470*, 2021.
- Vishal Gaur and Dorothée Honhon. Assortment planning and inventory decisions under a locational choice model. *Management Science*, 52(10):1528–1543, 2006.
- Jacob Goldin and Daniel Reck. Preference identification under inconsistent choice. *Available at SSRN*, 2015.
- Negin Golrezaei, Hamid Nazerzadeh, and Paat Rusmevichientong. Real-time optimization of personalized assortments. *Management Science*, 60(6):1532–1551, 2014.
- Negin Golrezaei, Vahideh Manshadi, Jon Schneider, and Shreyas Sekar. Learning product rankings robust to fake users. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 560–561, 2021.
- Negin Golrezaei, Vahideh Manshadi, Jon Schneider, and Shreyas Sekar. Learning product rankings robust to fake users. *Operations Research*, 2022.
- Alwin Haensel and Ger Koole. Estimating unconstrained demand rate functions using customer choice sets. *Journal of Revenue and Pricing Management*, 10(5):438–454, 2011.
- Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. ISSN 01621459. URL <http://www.jstor.org/stable/2282952>.
- Dorothee Honhon, Sreelata Jonnalagedda, and Xiajun Amy Pan. Optimal algorithms for assortment selection under ranking-based consumer choice models. *Manufacturing & Service Operations Management*, 14(2):279–289, 2012.
- Srikanth Jagabathula and Paat Rusmevichientong. A nonparametric joint assortment and price choice model. *Management Science*, 63(9):3128–3145, 2017.
- Srikanth Jagabathula and Paat Rusmevichientong. The limit of rationality in choice modeling: Formulation, computation, and implications. *Management Science*, 65(5):2196–2215, 2019.
- Srikanth Jagabathula and Gustavo Vulcano. A partial-order-based model to estimate individual preferences using panel data. *Management Science*, 64(4):1609–1628, 2018.



- Srikanth Jagabathula, Dmitry Mitrofanov, and Gustavo Vulcano. Personalized retail promotions through a directed acyclic graph-based representation of customer preferences. *Operations Research*, 70(2):641–665, 2022.
- Toshihiro Kamishima. Nantonac collaborative filtering: recommendation based on order responses. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 583–588, 2003.
- Toshihiro Kamishima, Hideto Kazawa, and Shotaro Akaho. Supervised ordering-an empirical survey. In *Fifth IEEE International Conference on Data Mining (ICDM’05)*, pages 4–pp. IEEE, 2005.
- David Kempe and Mohammad Mahdian. A cascade model for externalities in sponsored search. In *International Workshop on Internet and Network Economics*, pages 585–596. Springer, 2008.
- Minjeong Kim and Sharron J. Lennon. Consumer response to online apparel stockouts. *Psychology & Marketing*, 28(2):115–144, 2011.
- Jon Kleinberg and Eva Tardos. *Algorithm design*. Pearson Education India, 2006.
- Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.
- Wentao Lu, Ozge Sahin, and Ruxian Wang. A simple way towards fair assortment planning: Algorithms and welfare implications. *Available at SSRN 4514495*, 2023.
- Will Ma. When is assortment optimization optimal? *Management Science*, 69(4):2088–2105, 2023.
- Siddharth Mahajan and Garrett J. van Ryzin. Retail inventories and consumer choice. In *Quantitative models for supply chain management*, pages 491–551. Springer, 1999.
- Daniel McFadden. Conditional logit analysis of qualitative choice behavior. In Paul Zarembka, editor, *Frontiers in Econometrics*, pages 105–142. Academic Press, 1973.
- Daniel McFadden, Antti Talvitie, Stephen Cosslett, Ibrahim Hasan, Michael Johnson, Fred Reid, and Kenneth Train. Demand model estimation and validation. *Urban Travel Demand Forecasting Project, Phase, 1*, 1977.
- Jeffrey I. McGill and Garrett J. Van Ryzin. Revenue management: Research overview and prospects. *Transportation science*, 33(2):233–256, 1999.
- Rad Niazadeh, Negin Golrezaei, Joshua Wang, Fransisca Susan, and Ashwinkumar Badanidiyuru. Online learning via offline greedy: Applications in market design and optimization. *Available at SSRN 3613756*, 2020.
- Rad Niazadeh, Negin Golrezaei, Joshua R. Wang, Fransisca Susan, and Ashwinkumar Badanidiyuru. Online learning via offline greedy algorithms: Applications in market design and optimization. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 737–738, 2021.
- Alice Paul, Jacob Feldman, and James Mario Davis. Assortment optimization and pricing under a nonparametric tree choice model. *Manufacturing & Service Operations Management*, 20(3):550–565, 2018.
- Georgia Perakis and Divya Singhvi. Dynamic pricing with unknown non-parametric demand and limited price changes. *Available at SSRN 3336949*, 2019.

- Paat Rusmevichientong and Huseyin Topaloglu. Robust assortment optimization in revenue management under the multinomial logit choice model. *Operations research*, 60(4):865–882, 2012.
- Paat Rusmevichientong, Zuo-Jun Max Shen, and David B. Shmoys. Dynamic assortment optimization with a multinomial logit choice model and capacity constraint. *Operations research*, 58(6):1666–1680, 2010.
- Garrett van Ryzin and Siddharth Mahajan. On the relationship between inventory costs and variety benefits in retail assortments. *Management Science*, 45(11):1496–1509, 1999.
- Denis Sauré and Assaf Zeevi. Optimal dynamic assortment planning with demand learning. *Manufacturing & Service Operations Management*, 15(3):387–404, 2013.
- Burr Settles. Active learning literature survey. Technical report, Department of Computer Sciences, University of Wisconsin, Madison, 2009.
- Zhongshun Shi, Yijie Peng, Leyuan Shi, Chun-Hung Chen, and Michael C. Fu. Dynamic sampling allocation under finite simulation budget for feasibility determination. *INFORMS Journal on Computing*, 2021.
- Dongwook Shin, Mark Broadie, and Assaf Zeevi. Tractable sampling strategies for ordinal optimization. *Operations Research*, 66(6):1693–1712, 2018.
- A Serdar Şimşek and Huseyin Topaloglu. An expectation-maximization algorithm to estimate the parameters of the markov chain choice model. *Operations Research*, 66(3):748–760, 2018.
- Kalyan Talluri and Garrett Van Ryzin. Revenue management under a general discrete choice model of consumer behavior. *Management Science*, 50(1):15–33, 2004.
- Garrett van Ryzin and Gustavo Vulcano. A market discovery algorithm to estimate a general class of nonparametric choice models. *Management Science*, 61(2):281–300, 2015.
- Garrett van Ryzin and Gustavo Vulcano. An expectation-maximization method to estimate a rank-based choice model of demand. *Operations Research*, 65(2):396–407, 2017.
- Chieh-Hua Wen and Frank S. Koppelman. The generalized nested logit model. *Transportation Research Part B: Methodological*, 35(7):627–641, 2001.
- Gongbo Zhang, Haidong Li, and Yijie Peng. Sequential sampling for a ranking and selection problem with exponential sampling distributions. In *2020 Winter Simulation Conference (WSC)*, pages 2984–2995. IEEE, 2020.
- Zhiqiang Zheng and Balaji Padmanabhan. Selectively acquiring customer information: A new data acquisition problem and an active learning-based solution. *Management Science*, 52(5):697–712, 2006.

## Appendix

### A. Shorter Proofs Omitted in Main Body

#### A.1. Proof of Theorem 1

**THEOREM 1 (Impossibility Result).** *Suppose that the set of frequent rankings  $\Pi^F$  contains two rankings  $\pi$  and  $\pi'$  that are indistinguishable. Then, it is information-theoretically impossible to discover the set of types  $\Pi^F$  uniquely.*

**Proof.** Assume that  $\pi$  and  $\pi'$  are  $i$ -indistinguishable for  $2 \leq i \leq n - 2$ . We write  $q = p(\pi)$  and  $q' = p(\pi')$ . Assume that  $q \leq q'$ . We write  $A = \{\pi(1), \dots, \pi(i)\}$  for the set of the top- $i$  items common to  $\pi$  and  $\pi'$ . Let  $\bar{\pi}$  be the ranking that is the same as  $\pi$  in the first  $i$  positions and the same as  $\pi'$  in the rest. Similarly, let  $\bar{\pi}'$  be the ranking that is the same as  $\pi'$  in the first  $i$  positions and the same as  $\pi$  in the rest, i.e.  $\bar{\pi} = (\pi(1), \dots, \pi(i), \pi'(i+1), \dots, \pi'(n))$  and  $\bar{\pi}' = (\pi'(1), \dots, \pi'(i), \pi(i+1), \dots, \pi(n))$ . Notice that because of the definition of indistinguishability,  $\bar{\pi} \neq \bar{\pi}'$  and  $\{\bar{\pi}, \bar{\pi}'\} \cap \{\pi, \pi'\} = \emptyset$ .

We define the following alternate choice model probabilities  $\bar{p}(\cdot)$ :  $\bar{p}(\bar{\pi}) = \bar{p}(\bar{\pi}') = q$ ,  $\bar{p}(\pi') = q' - q$ , and  $\bar{p}(\pi) = 0$ . For all other rankings  $\pi'' \in \Pi$ , we set  $\bar{p}(\pi'') = p(\pi'')$ . As a result, depending on whether  $q' - q < \kappa$  or  $q' - q \geq \kappa$ , we obtain that  $\bar{\Pi}^F = (\Pi^F \setminus \{\pi, \pi'\}) \cup \{\bar{\pi}, \bar{\pi}'\}$ , or  $\bar{\Pi}^F = (\Pi^F \setminus \{\pi\}) \cup \{\bar{\pi}, \bar{\pi}'\}$ .

We will prove that no algorithm — regardless of computational power or number of queries — can distinguish between  $\Pi^F$  and  $\bar{\Pi}^F$ . We do so by a simple coupling argument: we couple the draws of types from  $\Pi^F$  and  $\bar{\Pi}^F$  in such a way that the algorithm will see the exact same responses for each query. Importantly, because the algorithm chooses the query *without knowledge of the consumer's type*, the coupling can be based on the query.

The coupling is straightforward. For any type  $\pi'' \in \Pi^F \setminus \{\pi, \pi'\}$ , the type  $\pi''$  is drawn under  $\Pi^F$  exactly when it is drawn under  $\bar{\Pi}^F$ . In particular, exactly the same item is chosen by the drawn consumer under both models, because the consumer has the same type.

For the remaining types  $\pi, \pi', \bar{\pi}, \bar{\pi}'$ , we use the following couplings, depending on the query set  $S$ :

- If  $S \cap A \neq \emptyset$ , then the coupling produces  $(\pi, \bar{\pi})$  with probability  $q$ ,  $(\pi', \bar{\pi}')$  with probability  $q$ , and  $(\pi', \pi')$  with probability  $q' - q$ . That is, for example, when  $\pi$  is generated under  $\Pi^F$  with probability  $q$ ,  $\bar{\pi}$  is also generated under  $\bar{\Pi}^F$  with the same probability.

Notice that the respective marginal probabilities are all as prescribed by the choice models, so this is a valid coupling. Furthermore, because only an item from  $S \cap A$  can be chosen, the same item will be chosen for each possible pair of types  $(\pi, \bar{\pi}), (\pi', \bar{\pi}'), (\pi', \pi')$ , because for each pair, the two types comprising it agree on the ranking of the top  $i$  items.

- If  $S \cap A = \emptyset$ , then the coupling produces  $(\pi, \bar{\pi}')$  with probability  $q$ ,  $(\pi', \bar{\pi})$  with probability  $q$ , and  $(\pi', \pi')$  with probability  $q' - q$ . Again, notice that all marginal probabilities match those prescribed by the choice models, so the coupling is valid.

Because the chosen item will always be from  $S \setminus A$ , all possible pairs of types will choose the same item; this is because each such pair has the same rankings for  $S \setminus A$ .

We have thus produced a valid coupling under which exactly the same item is chosen under  $\Pi^F$  and  $\overline{\Pi}^F$ . In particular, this means that the marginal probabilities of choosing each item are the same under both choice models, so an algorithm will observe the exact same distribution under both models, and cannot distinguish them. ■

## A.2. Proof of Proposition 2

**PROPOSITION 2 (Computing Choice Probabilities from a DAG).** *Let  $(\Pi, \mathbf{p})$  be a choice model, and  $(V, E, \mathbf{p}, \mathbf{e})$  a DAG representation of  $(\Pi, \mathbf{p})$ . Let  $S \subseteq \mathcal{N}$  be an item set and  $z \in S$ . Then, the set of types choosing  $z$  from  $S$  and its associated probability are characterized by the following:*

$$\Theta_z(S) = \bigcup_{A: A \cap S = \emptyset} \Theta_{A \oplus z} \quad q_z(S) = \sum_{A: A \cap S = \emptyset} e_{A \oplus z}.$$

**Proof.** To prove the first identity (that  $\Theta_z(S) = \bigcup_{A: A \cap S = \emptyset} \Theta_{A \oplus z}$ ), we prove both inclusions. First, any type  $\pi$  choosing  $z$  from  $S$  ranks some set  $A \subseteq \mathcal{N} \setminus S$  ahead of  $z$ , i.e., belongs to some  $\Theta_{A \oplus z}$  with  $A \cap S = \emptyset$ . Conversely, every type  $\pi \in \Theta_{A \oplus z}$  ranks  $z$  ahead of all other items in  $S$ , and so chooses  $z$  from  $S$ .

To prove the probability identity, we apply the definition of  $q_z(S)$ , then the set identity we just proved, then the disjointness of the sets  $\Theta_{A \oplus z}$  due to Proposition 1, and finally the definition of  $e_{A \oplus z}$ , to obtain that

$$q_z(S) = p(\Theta_z(S)) = p\left(\bigcup_{A: A \cap S = \emptyset} \Theta_{A \oplus z}\right) = \sum_{A: A \cap S = \emptyset} p(\Theta_{A \oplus z}) = \sum_{A: A \cap S = \emptyset} e_{A \oplus z}.$$

## A.3. Proof of Theorem 2

**THEOREM 2 (Constructing the Exact DAG using Exact Choice Probabilities).** *Let  $(\Pi, \mathbf{p})$  be the underlying choice model with  $T$  types, and assume that Algorithm 1 has exact oracle access to the choice probabilities. Then, Algorithm 1 runs in time  $O(n^2 T)$  and outputs a correct DAG representation of  $(\Pi, \mathbf{p})$ .*

**Proof.** Correctness follows directly by induction on the levels, showing that  $\widehat{p}(A) = p(A)$  for all  $A$ , and  $\widehat{e}_{A \oplus z} = e_{A \oplus z}$  for all  $A, z$ . The induction step uses Lemma 1 and Equation (3).

For the running time (and number of oracle queries), note that both the number of nodes at any level  $j$  as well as the number of edges from level  $j$  to  $j+1$  are upper-bounded by  $T$ .<sup>13</sup> To find all edges emanating from level  $j$ , the algorithm checks at most  $nT$  candidates (each possible item to add). Thus, the total computation at each level is at most  $O(nT)$ , and because there are  $n$  levels, the total computation takes  $O(n^2 T)$ . ■

<sup>13</sup> This upper bound could be quite loose. It is easy to construct cases in which the number of nodes is linear in  $n$ , while the number of types is exponential. In fact, this type of example illustrates the value of the DAG representation in compressing the model's description, and thereby making it more intuitively understandable.

## B. Set Cover and the Greedy Algorithm

The minimum set covering problem we define in Section 7.2 is equivalent to the classic SET COVER problem. In the classic SET COVER problem, a universe  $U = \{1, 2, \dots, n\}$  is given along with a collection  $\mathcal{L}$  of  $m$  sets whose union covers  $U$ . The goal is to find the smallest subcollection of  $\mathcal{L}$  whose union still covers  $U$ .

In our case, the universe is a subset  $\mathcal{P} \subseteq \mathcal{P}(A, z)$  because as can be seen in Lemma 2, the goal is to find a set cover of such sets. Since every element of  $\mathcal{P}(A, z)$  is a proper subset of  $A$ , for each element  $A' \in \mathcal{P}$ , there always exists an  $x$  such that  $x \in A \setminus A'$  and  $A' \subseteq A \setminus \{x\}$ . We remark that the reduction also works in the other direction: every SET COVER instance can be encoded in our setting. Given an instance  $(U, \mathcal{L})$ , we let  $j = |\mathcal{L}|$  be the number of sets, and set  $A = \{1, 2, \dots, j\}$ . Then, for each element  $u \in U$ , let  $T_u$  be the indices of sets  $S_i \in \mathcal{L}$  such that  $u \in S_i$ . We define a corresponding prefix  $A'_u = A \setminus T_u$ . Because  $T_u \neq \emptyset$  for all  $u$  (otherwise,  $u$  could never be covered), all  $A'_u$  are strict subsets of  $A$ . Furthermore,  $A'_u \subseteq A \setminus \{x\}$  for exactly those  $x$  such that  $u \in S_x$ ; thus, the instance we generated exactly encodes the original SET COVER instance.

The preceding reduction from SET COVER to our problem shows that both the NP-hardness and approximation hardness to within better than  $(1 - o(1)) \cdot \ln n$  (Feige (1998)) carry over to our problem. On the positive side, the reduction from our problem to SET COVER shows that approximation algorithms for the latter can be applied to our problem with the same guarantees. In particular, this applies to the well known greedy approximation algorithm (see, e.g., (Kleinberg and Tardos 2006)), which has a multiplicative approximation guarantee of  $\ln n$ .<sup>14</sup>

The greedy algorithm for SET COVER, denoted by GREEDYMINCOVER  $(U, \mathcal{L})$ , repeatedly adds a set  $L \in \mathcal{L}$  which covers the largest number of elements from  $U$  which had not been covered by previously added sets. It terminates when all of  $U$  is covered. Specialized to our setting, the greedy algorithm GREEDYMINCOVER  $(\mathcal{P}, \mathcal{S}^-(A))$  repeatedly selects a set from  $\mathcal{S}^-(A)$  that contains as subsets the largest number of prefixes  $A' \in \mathcal{P}$  not contained in any selected set so far. The algorithm terminates when all prefixes in  $\mathcal{P}$  have been covered.

## C. Proof of Proposition 3, and a More General Inclusion-Exclusion Lemma

Proposition 3 follows as an immediate corollary of the following lemma. (The more general lemma is also central to our analysis of ALGDAG.)

**LEMMA 2.** Fix a set  $A$  and item  $z \notin A$ . Let  $\mathcal{P} \subseteq \mathcal{P}(A, z)$ , and  $\mathcal{C}$  be a set cover of  $\mathcal{P}$  using  $\mathcal{S}^-(A)$ , with the additional property that  $\mathcal{P}$  is maximal, in the sense that  $\mathcal{C}$  is not a set cover of any  $\mathcal{P}' \supsetneq \mathcal{P}$ ,  $\mathcal{P}' \subseteq \mathcal{P}(A, z)$ . Then,

$$\sum_{A' \in \mathcal{P}} e_{A' \oplus z} = \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C).$$

<sup>14</sup> There are other known approximation algorithms for SET COVER with the same guarantee.

**Proof.** We begin by observing that because  $\mathcal{C}$  is a set cover of  $\mathcal{P}$ , every  $A' \in \mathcal{P}$  is contained in some  $C \in \mathcal{C}$ . As a result,  $\Theta_{A' \oplus z} \subseteq \Theta_z(C)$ ; this is because any ranking in  $\Theta_{A' \oplus z}$  prefers  $z$  over all of  $\mathcal{N} \setminus A' \supseteq \mathcal{N} \setminus C$ , and will therefore choose  $z$  from  $\mathcal{N} \setminus C$ . We have thus shown that  $\bigcup_{A' \in \mathcal{P}} \Theta_{A' \oplus z} \subseteq \bigcup_{C \in \mathcal{C}} \Theta_z(\mathcal{N} \setminus C)$ . For the converse inclusion, observe that for each type  $\pi \in \Theta_z(\mathcal{N} \setminus C)$ , the set of items  $A'$  ranked ahead of  $z$  in  $\pi$  must be a subset of  $C$ . Therefore,  $\pi \in \Theta_{A' \oplus z}$  for this  $A'$ . Because  $A' \in \mathcal{P}(A, z)$ , my maximality of  $\mathcal{P}$ , we must have  $A' \in \mathcal{P}$  as well. Thus, we have shown the converse inclusion  $\bigcup_{A' \in \mathcal{P}} \Theta_{A' \oplus z} \supseteq \bigcup_{C \in \mathcal{C}} \Theta_z(\mathcal{N} \setminus C)$ . Applying that the sets  $\Theta_{A' \oplus z}$  are all disjoint by Proposition 1, we obtain that

$$\sum_{A' \subsetneq A} e_{A' \oplus z} = p \left( \bigcup_{A' \subsetneq A} \Theta_{A' \oplus z} \right) = p \left( \bigcup_{C \in \mathcal{C}} \Theta_z(\mathcal{N} \setminus C) \right).$$

In the remainder of the proof, we will show that  $p \left( \bigcup_{C \in \mathcal{C}} \Theta_z(\mathcal{N} \setminus C) \right)$  equals the Inclusion-Exclusion sum. For any non-empty subset  $\mathcal{B} \subseteq \mathcal{C}$ , we write

$$S_{\mathcal{B}} = \mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C = \bigcup_{C \in \mathcal{B}} (\mathcal{N} \setminus C).$$

This implies that  $S_{\mathcal{B} \cup \mathcal{B}'} = S_{\mathcal{B}} \cup S_{\mathcal{B}'}$ . Because  $z \notin C$  for all  $C \in \mathcal{C}$  (because  $\mathcal{C} \subseteq \mathcal{S}^-(A)$ , and  $z \notin A$ ), we obtain that  $z \in S_{\mathcal{B}}$ . A type  $\pi$  chooses  $z$  from  $\bigcup_{C \in \mathcal{B}} (\mathcal{N} \setminus C)$  if and only if  $\pi$  chooses  $z$  from  $\mathcal{N} \setminus C$  for each  $C \in \mathcal{B}$ . Therefore,  $\Theta_z(S_{\mathcal{B}}) = \bigcap_{C \in \mathcal{B}} \Theta_z(\mathcal{N} \setminus C)$ . Applying the standard Inclusion-Exclusion formula to the sets  $\Theta_z(\mathcal{N} \setminus C)$ , we obtain that

$$\begin{aligned} p \left( \bigcup_{C \in \mathcal{C}} \Theta_z(\mathcal{N} \setminus C) \right) &= \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot p \left( \bigcap_{C \in \mathcal{B}} \Theta_z(\mathcal{N} \setminus C) \right) \\ &= \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot p(\Theta_z(S_{\mathcal{B}})) \\ &= \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z \left( \mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C \right), \end{aligned}$$

where the last step used that  $q_z(S) = p(\Theta_z(S))$  for all sets  $S \ni z$ . ■

**Proof of Proposition 3.** Applying Lemma 2 with  $\mathcal{P} = \mathcal{P}(A, z)$  gives us that  $\sum_{A' \subsetneq A} e_{A' \oplus z} = \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C)$ . Substituting this identity into Lemma 1 and rearranging now completes the proof. ■

## D. Proof of Theorem 3

In this section, we give a proof of Theorem 3, with the proofs of two technical lemmas deferred to separate appendices below.

The main part of the analysis is captured by Lemma 4, which will be stated later in this section. This lemma shows that under the general model, the estimates  $\hat{e}_{A \oplus z}$  which ALGIE obtains for the edge probabilities  $e_{A \oplus z}$  are accurate enough with sufficiently high probability. The lemma further bounds the number of

queries required under the  $(\kappa, \rho)$ -Random Model. The number of queries required increases exponentially in the size of the set covers used, so to bound the former, we would like to bound the latter. We observe that only certain configurations of types could possibly cause a need for large set covers. Therefore, we first show that under the  $(\kappa, \rho)$ -Random Model, these configurations are sufficiently unlikely to occur.

More formally, for any positive integer  $c \in [n - 1]$ , we define  $\mathcal{E}_c$  to be the event that there are two distinct frequent types  $\pi_1 \neq \pi_2 \in \Pi^F$ , a position  $j \leq n_0$ , and an item  $z$  in position  $j + 1$  of ranking  $\pi_1$ , such that

- $z$  is in position  $c + 1$  or later in  $\pi_2$ , and
- the set of items preceding  $z$  in  $\pi_2$  is a subset of the first  $j$  items in  $\pi_1$ .

Lemma 3, proved in Appendix E, shows that  $\mathcal{E}_c$  is unlikely to occur.

**LEMMA 3.** *Under the  $(\kappa, \rho)$ -Random Model, when  $c \geq \log_{1/\alpha} \left( \frac{K^2 n}{\delta} \right)$ , the probability of  $\mathcal{E}_c$  is at most  $\mathbb{P}[\mathcal{E}_c] \leq \delta$ .*

Because  $\mathcal{E}_c$  can never happen for  $c > n$ , Lemma 3 is of interest only in the regime when  $\log_{1/\alpha} \left( \frac{K^2 n}{\delta} \right) \leq n$ , or  $K \leq \sqrt{\frac{(1/\alpha)^n \delta}{n}}$ . Otherwise, the lemma is trivial since the probability of the event  $\mathcal{E}_c$  is 0.

Next, we show that having computed the graph correctly up to the previous level is enough to ensure that ALGIE returns accurate estimates with high probability — this forms the key of the inductive correctness proof of ALGDAG. Furthermore, when  $\mathcal{E}_c$  happens, all the set cover solutions in ALGIE are small enough that the number of samples stays polynomial.

**LEMMA 4.** *Let  $j \in \{0, 1, \dots, n_0 - 1\}$  be a level, and assume that  $\widehat{V}_j = V_j^F$  and  $\widehat{E}_j = E_j^F$ . Let  $\widehat{e}_{A \oplus z}$  be the estimated edge probability returned by ALGIE (Algorithm 3), called with parameters  $\epsilon$  and  $\delta$ . Then,  $\widehat{e}_{A \oplus z}$  satisfies*

$$|\widehat{e}_{A \oplus z} - e_{A \oplus z}| \leq \epsilon + \rho$$

with probability at least  $1 - \delta$ .

Furthermore, if  $\mathcal{E}_c$  did not happen, then the number of consumer queries and the total computation time are upper-bounded by  $O \left( \frac{2^{2c} (c + \log(1/\delta))}{\epsilon^2} \right)$ .

**Proof of Theorem 3.** We are now ready to prove Theorem 3. We will prove by induction on the level  $j$  that with probability at least  $(1 - \frac{\delta}{n_0})^j$ , the following four all hold:

$$\widehat{V}_j = V_{n_0}^F \quad \widehat{E}_j = E_{n_0}^F \quad \max_{A \in V_j} |\widehat{p}(A) - p(A)| \leq \epsilon + n_0 \rho \quad \max_{A \in V_j, A \cup \{z\} \in V_j} |\widehat{e}_{A \oplus z} - e_{A \oplus z}| \leq \epsilon / n_0 + \rho.$$

The first part of the theorem then follows by setting  $j = n_0 = \alpha n$ , and noting that  $(1 - \frac{\delta}{n_0})^{n_0} \geq 1 - \delta$ .

We now complete the induction proof. The base case  $j = 0$  holds because  $\widehat{V}_0 = \{\emptyset\} = V_0^F$  and  $\widehat{E}_0 = \emptyset = E_0^F$  hold deterministically. For the induction step, consider a  $j < n_0$ . By induction hypothesis,  $\widehat{V}_j = V_j^F$  and  $\widehat{E}_j = E_j^F$  with probability at least  $(1 - \frac{\delta}{n_0})^j$  (and the estimates of  $\mathbf{p}$  and  $\mathbf{e}$  up to level  $j$  are accurate to within an additive  $\epsilon + n_0 \rho$  and  $\epsilon + \rho$ , respectively). We condition on this case.

To obtain  $\widehat{G}_{j+1}$  from  $\widehat{G}_j$ , Algorithm 2 goes through each node  $A$  on level  $j$  of  $\widehat{G}_j$  and all candidate items  $z \notin A$ , and uses ALGIE to compute an estimate  $\widehat{e}_{A \oplus z}$  of the true value  $e_{A \oplus z}$ . Because  $\widehat{V}_j = V_j^F$  and  $\widehat{E}_j = E_j^F$ , Lemma 4 implies that for each of these calls (characterized by the choice of  $z$ ), ALGIE (which is called by ALGDAG with parameters  $\epsilon' = \min(\epsilon/n_0, \kappa/4)$  and  $\delta' = \frac{\delta}{\alpha n^2 K}$ ) returns an estimate  $\widehat{e}_{A \oplus z}$  that is at most  $\epsilon' + \rho = \min(\epsilon/n_0, \kappa/4) + \rho$  far from the true value  $e_{A \oplus z}$ , with probability at least  $1 - \delta' = 1 - \frac{\delta}{\alpha n^2 K}$ .

Because there are at most  $K$  frequent types, and the nodes on level  $j$  of  $\widehat{G}_j$  correspond to disjoint sets of types, there are at most  $K$  nodes on level  $j$ . For each such node, the algorithm checks at most  $n$  items  $z$  to add. By a union bound over all of the invocations of ALGIE for these different pairs  $(A, z)$ , all estimates  $\widehat{e}_{A \oplus z}$  of  $e_{A \oplus z}$  are simultaneously accurate to within an additive  $\epsilon' + \rho$  with probability at least

$$\left(1 - \frac{\delta}{\alpha n^2 K}\right)^{nK} \geq 1 - (nK) \cdot \frac{\delta}{\alpha n^2 K} = 1 - \frac{\delta}{\alpha n} = 1 - \frac{\delta}{n_0}.$$

Under the assumptions of the theorem,  $\epsilon' + \rho \leq \kappa/4 + \rho < \kappa/2$ . In particular, when the estimates are accurate to within  $\epsilon' + \rho$ , rare rankings alone can never make the algorithm add a node  $A \cup \{z\}$  to layer  $j + 1$ . Conversely, every frequent type has probability at least  $\kappa$ , so when the estimates are accurate to within  $\epsilon' + \rho$ , every frequent type whose first  $j + 1$  items are  $A \cup \{z\}$  has estimated frequency at least  $\kappa - (\epsilon' + \rho) \geq \kappa - (\kappa/4 + \rho) \geq \kappa/2$ , hence causes the creation of the node  $A \cup \{z\}$ . Thus, we have shown that  $\widehat{V}_{j+1} = V_{j+1}^F$  and  $\widehat{E}_{j+1} = E_{j+1}^F$  whenever all estimates are accurate to within  $\epsilon' + \rho$ .

To bound the error in the estimated probabilities, first observe that the bound on  $|\widehat{e}_{A \oplus z} - e_{A \oplus z}|$  follows directly because  $\epsilon' + \rho \leq \epsilon/n_0 + \rho$ . Now, consider a node  $A \in V_{j+1}^F$ . The true combined probability of types who rank all of  $A$  ahead of all items not in  $A$  is  $p(A) = \sum_{z \in A} e_{A \setminus \{z\} \oplus z}$ . ALGDAG, on the other hand, uses estimated probabilities  $\widehat{e}_{A \setminus \{z\} \oplus z}$  in place of the true probabilities, and furthermore only adds these estimates for elements  $z$  such that the set  $A \setminus \{z\} \in V_j^F$  is a frequent type, i.e., it uses  $\widehat{p}(A) = \sum_{z \in A: A \setminus \{z\} \in V_j^F} \widehat{e}_{A \setminus \{z\} \oplus z}$ . Using the triangle inequality along with the upper bound of  $\epsilon' + \rho$  on individual estimation errors, and the fact that  $|A| \leq n_0 - 1$ , we get that the error is at most

$$\begin{aligned} |p(A) - \widehat{p}(A)| &= \left| \sum_{z \in A} e_{A \setminus \{z\} \oplus z} - \sum_{z \in A: A \setminus \{z\} \in V_j^F} \widehat{e}_{A \setminus \{z\} \oplus z} \right| \\ &\leq \sum_{z \in A: A \setminus \{z\} \in V_j^F} |e_{A \setminus \{z\} \oplus z} - \widehat{e}_{A \setminus \{z\} \oplus z}| + \sum_{z \in A: A \setminus \{z\} \notin V_j^F} e_{A \setminus \{z\} \oplus z} \\ &\leq \sum_{z \in A: A \setminus \{z\} \in V_j^F} (\min(\epsilon/n_0, \kappa/4) + \rho) + \rho \\ &\leq n_0 \cdot \frac{\epsilon}{n_0} + (n_0 - 1) \cdot \rho + \rho \\ &= \epsilon + n_0 \cdot \rho. \end{aligned}$$

Here, the inequality on the third line used the fact that the last sum adds probabilities that can only arise from disjoint sets of rare types.



Finally, by induction hypothesis, the event we conditioned on (that  $\widehat{V}_j = V_j^F$  and  $\widehat{E}_j = E_j^F$  and the estimates up to level  $j$  are accurate to within an additive  $\epsilon + n_0\rho$ ) had probability at least  $(1 - \frac{\delta}{n_0})^j$ , and conditioned on this event, level  $j + 1$  is correct (and accurate to within the claimed additive errors) with probability at least  $(1 - \frac{\delta}{n_0})$ . Hence, the entire inferred DAG representation up to level  $j + 1$  is accurate with probability at least  $(1 - \frac{\delta}{n_0})^{j+1}$ , completing the induction step, and thus the proof of the first part of the theorem.

We now prove the second part of the theorem, bounding the number of queries and the computation time under the random model. Define  $c = \lceil \log_{1/\alpha} \left( \frac{K^2 n}{\delta} \right) \rceil$ . By Lemma 3, with probability at least  $1 - \delta$ , the event  $\mathcal{E}_c$  did not happen. By a union bound with the first part of the proof, we obtain that with probability at least  $1 - 2\delta$ ,  $G_{n_0}^F$  is correctly reconstructed, all probability estimates are accurate to within the claimed additive errors, and the event  $\mathcal{E}_c$  did not happen.

By the second part of Lemma 4, each invocation of ALGIE results in  $O\left(\frac{K^{4\nu} \cdot n^{2\nu} \cdot \log(nK/\delta)}{\delta^{2\nu} \cdot \min(\epsilon, \kappa)^2}\right)$  consumer queries and computation. Each frequent type contributes at most one distinct node  $A$  for each level  $j$ , and for each such node  $A$ , ALGDAG considers all items  $z \notin A$  for addition, and calls ALGIE. Thus, the total number of calls to ALGIE is at most  $Knn_0$ . Substituting, bounding that  $n_0 \leq n$ , and writing the constant  $\nu = \log_\alpha(1/2) > 0$ , we obtain that the number of consumer queries is at most

$$O\left(Knn_0 \cdot \frac{2^{2c}(c + \log(2\alpha n^2 K/\delta))}{\min(\epsilon/n_0, \kappa/4)^2}\right) = O\left(\frac{K^{1+4\nu} \cdot n^{4+2\nu} \cdot \log(nK/\delta)}{\delta^{2\nu} \cdot \min(\epsilon, \kappa)^2}\right),$$

which is polynomial. Since the computation is dominated by the queries, the same bound applies to the computation time. ■

## E. Proof of Lemma 3

In this section, we prove Lemma 3, which is restated below.

**LEMMA 3.** *Under the  $(\kappa, \rho)$ -Random Model, when  $c \geq \log_{1/\alpha} \left( \frac{K^2 n}{\delta} \right)$ , the probability of  $\mathcal{E}_c$  is at most  $\mathbb{P}[\mathcal{E}_c] \leq \delta$ .*

**Proof.** We want to bound the probability that there are two distinct types  $\pi_1 \neq \pi_2 \in \Pi^F$  and a position  $j \leq n_0$  such that the item  $z$  which is in position  $j + 1$  in  $\pi_1$  is in position  $c + 1$  or later in  $\pi_2$ , and the set of items preceding  $z$  in  $\pi_2$  is a subset of the first  $j$  items in  $\pi_1$ .

For now, focus on two types and a fixed position  $j \leq n_0$ , with  $\pi_1$  already drawn (and thus defining  $z$ ), and the uniformly random draw defining  $\pi_2$ . Let  $\mathcal{E}_{c,k}$  (for  $k \geq c + 1$ ) denote the event that  $z$  is in position  $k$  in  $\pi_2$ , and the first  $k - 1$  items of  $\pi_2$  are all among the first  $j$  items of  $\pi_1$ .

There are  $\binom{j}{k-1}$  ways to pick the first  $k - 1$  items of  $\pi_2$  from the first  $j$  items of  $\pi_1$ ,  $(k - 1)!$  ways to order them, and  $(n - k)!$  ways to order the items after position  $k$ . Since there are  $n!$  total rankings, the probability of  $\mathcal{E}_{c,k}$  is

$$\mathbb{P}[\mathcal{E}_{c,k}] = \frac{\binom{j}{k-1} \cdot (k-1)! \cdot (n-k)!}{n!} = \frac{j! \cdot (n-k)!}{(j+1-k)! \cdot n!} \leq \frac{j! \cdot (n-(c+1))!}{(j-c)! \cdot n!},$$

where the inequality holds because the probability is monotone decreasing in  $k$  and thus maximized when  $k$  is as small as possible, i.e.,  $k = c + 1$ . Next, we bound

$$\frac{j! \cdot (n - (c + 1))!}{(j - c)! \cdot n!} = \frac{1}{n - c} \cdot \prod_{k=0}^{c-1} \frac{j - k}{n - k} \leq \frac{1}{n - c} \cdot (j/n)^c \leq \frac{1}{n - c} \cdot (n_0/n)^c,$$

because  $(j - k)/(n - k)$  is monotone decreasing in  $k$ , so it is maximized at  $k = 0$ , and  $j \leq n_0$ . Now, taking a union bound over all choices of  $k = c + 1, \dots, j$  (of which there are at most  $n - c$ ) as well as all choices of  $j \leq n_0 \leq n$  and ordered pairs of types (of which there are at most  $K(K - 1) \leq K^2$ ), we obtain the bound

$$\mathbb{P}[\mathcal{E}_c] \leq K^2 \cdot n \cdot (n - c) \cdot \frac{1}{n - c} \cdot (n_0/n)^c = K^2 \cdot n \cdot \alpha^c.$$

Finally, substituting the lower bound on  $c$ , we obtain that

$$\mathbb{P}[\mathcal{E}_c] \leq K^2 \cdot n \cdot \alpha^c \leq K^2 \cdot n \cdot \frac{\delta}{K^2 \cdot n} = \delta,$$

completing the proof. ■

## F. Proof of Lemma 4

In this section, we complete the proof of Lemma 4, which is restated below for convenience.

**LEMMA 4.** *Let  $j \in \{0, 1, \dots, n_0 - 1\}$  be a level, and assume that  $\widehat{V}_j = V_j^F$  and  $\widehat{E}_j = E_j^F$ . Let  $\widehat{e}_{A \oplus z}$  be the estimated edge probability returned by ALGIE (Algorithm 3), called with parameters  $\epsilon$  and  $\delta$ . Then,  $\widehat{e}_{A \oplus z}$  satisfies*

$$|\widehat{e}_{A \oplus z} - e_{A \oplus z}| \leq \epsilon + \rho$$

with probability at least  $1 - \delta$ .

Furthermore, if  $\mathcal{E}_c$  did not happen, then the number of consumer queries and the total computation time are upper-bounded by  $O\left(\frac{2^{2c}(c + \log(1/\delta))}{\epsilon^2}\right)$ .

**Proof of Lemma 4.** First, by assumption of the lemma, we have that  $\widehat{V}_j = V_j^F$  and  $\widehat{E}_j = E_j^F$ , which implies that  $\widehat{\mathcal{P}} = \mathcal{P}^F(A, z)$  is the set of all prefixes  $A' \subsetneq A$  such that at least one frequent type has  $A'$  in the first  $|A'|$  positions, followed by  $z$ . Recall that ALGIE computes an approximately smallest set cover  $\mathcal{C}$  of  $\widehat{\mathcal{P}} = \mathcal{P}^F(A, z)$  using sets in  $\mathcal{S}^-(A)$ ; specifically, this means that  $\mathcal{C} \subseteq \mathcal{S}^-(A)$  and that for each set  $A' \in \mathcal{P}^F(A, z)$ , there exists a set  $C \in \mathcal{C}$  with  $A' \subseteq C$ .

By Lemma 1, the true probability of having a prefix of  $A$  followed by  $z$  is

$$e_{A \oplus z} = q_z(\mathcal{N} \setminus A) - \sum_{A' \subsetneq A} e_{A' \oplus z}.$$

Let  $\mathcal{P} \supseteq \mathcal{P}^F(A, z)$ ,  $\mathcal{P} \subseteq \mathcal{P}(A, z)$  be maximal such that  $\mathcal{C}$  is a set cover of  $\mathcal{P}$ . Then, by Lemma 2, applied to  $\mathcal{P}$ , we have that

$$\sum_{A' \in \mathcal{P}} e_{A' \oplus z} = \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C).$$

Substituting this identity into the previous identity for  $e_{A \oplus z}$ , we obtain that

$$\begin{aligned} \left| e_{A \oplus z} - \left( q_z(\mathcal{N} \setminus A) - \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right) \right| &= \sum_{A' \subsetneq A, A' \notin \mathcal{P}} e_{A' \oplus z} \\ &= p \left( \bigcup_{A' \subsetneq A, A' \notin \mathcal{P}} \Theta_{A' \oplus z} \right) \\ &\leq \rho. \end{aligned}$$

Here, the second step used the disjointness of the  $A'$  from Proposition 1, and the final step used that every type  $\pi \in \Theta_{A' \oplus z}$  with  $A' \subsetneq A, A' \notin \mathcal{P}$  must be a rare type, because  $\mathcal{P} \supseteq \mathcal{P}^F(A, z)$ .

ALGIE, on the other hand, uses the estimated  $\hat{q}_z$  values to compute an estimate of the edge probability

$$\hat{e}_{A \oplus z} = \hat{q}_z(\mathcal{N} \setminus A) - \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot \hat{q}_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C).$$

Thus, the absolute estimation error is

$$\begin{aligned} |e_{A \oplus z} - \hat{e}_{A \oplus z}| &= \left| e_{A \oplus z} - \left( q_z(\mathcal{N} \setminus A) - \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right) \right. \\ &\quad + \left( q_z(\mathcal{N} \setminus A) - \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right) \\ &\quad \left. - \left( \hat{q}_z(\mathcal{N} \setminus A) - \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot \hat{q}_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right) \right| \\ &\leq \left| e_{A \oplus z} - \left( q_z(\mathcal{N} \setminus A) - \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} (-1)^{|\mathcal{B}|+1} \cdot q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right) \right| \\ &\quad + |q_z(\mathcal{N} \setminus A) - \hat{q}_z(\mathcal{N} \setminus A)| + \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} \left| q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) - \hat{q}_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right| \\ &\leq \rho + |q_z(\mathcal{N} \setminus A) - \hat{q}_z(\mathcal{N} \setminus A)| + \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} \left| q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) - \hat{q}_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right|, \end{aligned}$$

where the first inequality is based on applying the triangle inequality first at the outer level, then pulling it through the sum.

The key step for the analysis is to upper-bound the differences  $|q_z(\mathcal{N} \setminus S) - \hat{q}_z(\mathcal{N} \setminus S)|$ , where either  $S = A$  or  $S = \bigcap_{C \in \mathcal{B}} C$  for some  $\mathcal{B} \subseteq \mathcal{C}$ . We do this using Hoeffding's Inequality (Lemma 5). Specifically,  $\hat{q}_z(\mathcal{N} \setminus S)$  is the average of at least

$$m = \left\lceil \frac{2^{2|\mathcal{C}|-1} \cdot (\ln(1/\delta) + (|\mathcal{C}| + 1) \cdot \ln(2))}{\epsilon^2} \right\rceil$$

i.i.d. Bernoulli random variables, with probability  $q_z(\mathcal{N} \setminus S)$  of taking the value 1. Hence, by Hoeffding's Inequality for  $m$  Bernoulli random variables with mean  $q_z(\mathcal{N} \setminus S)$ , and by setting  $\tau = \frac{\epsilon}{2^{|\mathcal{C}|}}$ , we have

$$\begin{aligned} \mathbb{P}[|q_z(\mathcal{N} \setminus S) - \hat{q}_z(\mathcal{N} \setminus S)| > \tau] &\leq 2 \exp(-2\tau^2 m) \\ &= 2 \exp \left( -2 \left( \frac{\epsilon}{2^{|\mathcal{C}|}} \right)^2 \left\lceil \frac{2^{2|\mathcal{C}|-1} \cdot (\ln(1/\delta) + (|\mathcal{C}| + 1) \cdot \ln(2))}{\epsilon^2} \right\rceil \right) \\ &\leq \frac{\delta}{2^{|\mathcal{C}|}}. \end{aligned}$$

There are  $2^{|\mathcal{C}|} - 1$  sets  $S$  of the form  $S = \bigcap_{C \in \mathcal{B}} C$  (one for each  $\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset$ ), plus the set  $S = A$ . Hence, by a union bound over all such sets  $S$ , with probability at least  $1 - 2^{|\mathcal{C}|} \cdot \frac{\delta}{2^{|\mathcal{C}|}} = 1 - \delta$ , all of the estimates satisfy  $|q_z(\mathcal{N} \setminus S) - \hat{q}_z(\mathcal{N} \setminus S)| \leq \frac{\epsilon}{2^{|\mathcal{C}|}}$ . In that case, the total estimation error is at most

$$\begin{aligned} |e_{A \oplus z} - \hat{e}_{A \oplus z}| &\leq \rho + |q_z(\mathcal{N} \setminus A) - \hat{q}_z(\mathcal{N} \setminus A)| + \sum_{\mathcal{B} \subseteq \mathcal{C}, \mathcal{B} \neq \emptyset} \left| q_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) - \hat{q}_z(\mathcal{N} \setminus \bigcap_{C \in \mathcal{B}} C) \right| \\ &\leq \rho + 2^{|\mathcal{C}|} \cdot \frac{\epsilon}{2^{|\mathcal{C}|}} \\ &= \rho + \epsilon. \end{aligned}$$

Finally, we prove the second part of the lemma, i.e., the bound on the number of consumer queries (and hence computation, since the inclusion-exclusion computation dominates the computing time as well). We will show that whenever  $\mathcal{E}_c$  did not happen,  $|\mathcal{C}| \leq c$ . For contradiction, assume that  $|\mathcal{C}| > c$ , i.e., the greedy min cover algorithm returned a solution of size at least  $c + 1$ . Recall that the greedy algorithm selects subsets of the form  $A \setminus \{x\}$  until all prefixes are covered, and that by the assumption of the lemma that  $\hat{G}_j = G_j^F$ , the set of prefixes that the greedy algorithm is trying to cover is exactly  $\mathcal{P}^F(A, z)$ . Consider the moment when the greedy algorithm has selected  $c$  sets  $C_i = A \setminus \{x_i\}$  in its cover. At that point, there is still an uncovered set  $A' \in \mathcal{P}^F(A, z)$ , since the greedy algorithm added at least one more set. Because  $A'$  is not contained in any of the  $C_i$ , it must contain all of the elements  $x_i$ ; in particular, it has size at least  $c$ . Furthermore, by definition of  $\mathcal{P}^F(A, z)$ ,  $A' \subseteq A$ , and  $A'$  is a prefix followed by  $z$ . Therefore, there exist a frequent type  $\pi_1$  which has all of  $A$  followed by  $z$  and a type  $\pi_2$  which has all of  $A'$  preceding  $z$  directly, where  $A' \subseteq A$  and  $|A'| \geq c$ , implying that  $z$  is in position  $c + 1$  or later in  $\pi_2$ . This certifies that the event  $\mathcal{E}_c$  happened, which the assumption of the lemma ruled out. Substituting the bound  $|\mathcal{C}| \leq c$  into the definition of  $m$  gives us the claimed bound.  $\blacksquare$

**LEMMA 5 (Hoeffding's Inequality (Hoeffding (1963))).** *Let  $X_1, X_2, \dots, X_n$  be independent random variables bounded between  $[0, 1]$ . Then,*

$$\mathbb{P} \left[ \left| \sum_i (X_i - \mathbb{E}[X_i]) \right| > \tau \right] \leq 2 \exp(-2\tau^2/n).$$

## G. Additional Details on Experiments

### G.1. Benchmark Implementation

For both synthetic and sushi data sets, we do the following for the benchmark: (1) solve the primal LP fully (without constraint sampling) using the Gurobi solver, instead of solving the dual approximately (as done in Farias et al. (2013)), and (2) impose sparsity on  $\lambda$ , the probability distribution of types. These give the primal LP algorithm an extra benefit: solving the primal fully permits searching over a bigger space, and the sparsity constraint helps the LP deal with the indistinguishability problem. Furthermore, we

relax the LP constraint to overcome infeasibility, by incorporating  $1/\sqrt{m}$  standard error for the empirically calculated probability that each item in the assortment is chosen, where  $m$  is the number of samples used per offered assortment in ALGDAG and the benchmark. So instead of having the constraint  $\mathbf{y} = A\boldsymbol{\lambda}$ , we have  $\hat{\mathbf{y}} - 1/\sqrt{m} \cdot \mathbf{1} \leq A\boldsymbol{\lambda} \leq \hat{\mathbf{y}} + 1/\sqrt{m} \cdot \mathbf{1}$ . We do so because the primal LP often does not yield any solutions under the constraint  $\hat{\mathbf{y}} = A\boldsymbol{\lambda}$  where  $\hat{\mathbf{y}}$  is empirically calculated using the same number of samples as ALGDAG. This is mainly because the system of equations can be overspecified, so if we use an estimated version of  $\mathbf{y}$  that might have some error, two or more systems of equations can result in different values, leading to a contradiction.

## G.2. Additional Experimental Results

As a robustness check on the performance of ALGDAG, for the synthetic setting, we vary the number of types and the number of items. Specifically, we consider a setting with 15 frequent types and 20 items. The total probability  $\rho$  of noise is still chosen from the set  $\{0, 0.001, 0.01, 0.05\}$ , and 20 noisy types are generated when  $\rho$  is greater than 0. The probabilities of frequent and noisy types are generated as those for the first synthetic setting (see Section 9), i.e., according to a symmetric Dirichlet distribution with coefficient of variation 0.1, multiplied by  $(1 - \rho)$  and  $\rho$ , respectively. The goal is to estimate the top  $n_0 \in \{5, 10\}$  positions, and we choose  $\kappa$  and  $\epsilon$  to be 0.01 and 0.001, respectively. Results are then averaged over 100 generated instances.

	Noise = 0		Noise = 0.001		Noise = 0.01		Noise = 0.05	
	$n_0 = 5$	$n_0 = 10$	$n_0 = 5$	$n_0 = 10$	$n_0 = 5$	$n_0 = 10$	$n_0 = 5$	$n_0 = 10$
max. prob. differences	0.0099± 0.00045	0.0102± 0.00049	0.0114± 0.00048	0.0159± 0.00050	0.0122± 0.00036	0.0157± 0.00046	0.0132± 0.00042	0.0160± 0.00059
% of different vertices	0.89%± 0.272%	1.01%± 0.242%	5.10%± 0.376%	7.68%± 0.388%	6.47%± 0.436%	8.97%± 0.370%	7.94%± 0.580%	9.98%± 0.502%

**Table 6** The maximum probability discrepancy over all nodes, between the DAG returned by ALGDAG and the true DAG, and the percentage of vertices that are added wrongly or not inferred by ALGDAG, averaged over 100 instances run on synthetic data with 15 frequent types and 20 items.

The experimental results for this setting are summarized in Table 6. The maximum probability discrepancy is 0.0160, which is obtained when the top 10 positions are being estimated. The value of this maximum probability discrepancy is comparable with that in the 5 frequent types setting. Similarly, the percentages of different vertices over different frequent cases are at most 9.98%, which is comparable with the result for the cases with 5 frequent types. The benchmark is not included in this setting because with a large number of items, the benchmark is too computationally inefficient.