Markov Layout

Flavio Chierichetti[*]
Department of Computer Science
Cornell University
Ithaca, NY 14853, USA

Ravi Kumar
Yahoo! Research
Sunnyvale, CA 94089, USA

Prabhakar Raghavan
Yahoo! Labs
Sunnyvale, CA 94089, USA

Abstract

Consider the following problem of laying out a set of n images that match a query onto the nodes of a $\sqrt{n} \times \sqrt{n}$ grid. We are given a score for each image, as well as the distribution of patterns by which a user's eye scans the nodes of the grid and we wish to maximize the expected total score of images selected by the user. This is a special case of the *Markov layout problem*, in which we are given a Markov chain M together with a set of objects to be placed at the states of the Markov chain. Each object has a utility to the user if viewed, as well as a stopping probability with which the user ceases to look further at objects. We point out that this layout problem is prototypical in a number of applications in web search and advertising, particularly in the emerging genre of search results pages from major engines. In a different class of applications, the states of the Markov chain are web pages at a publishers website and the objects are advertisements.

In this paper we study the approximability of the Markov layout problem. Our main result is an $O(\log n)$ approximation algorithm for the most general version of the problem. The core idea behind the algorithm is to transform an optimization problem over partial permutations into an optimization problem over sets by losing a logarithmic factor in approximation; the latter problem is then shown to be submodular with two matroid constraints, which admits a constant-factor approximation. In contrast, we also show the problem is APX-hard via a reduction from CUBIC MAX-BISECTION.

We then study harder variants of the problem in which no *gaps* — states of M with no object placed on them — are allowed. By exploiting the geometry, we obtain an $O(\log^{3/2} n)$ approximation algorithm when the digraph underlying M is a grid and an $O(\log n)$ approximation algorithm when it is a tree. These special cases are especially appropriate for our applications.

[*]Work done in part while visiting Yahoo! Research. Supported in part by NSF grant CCF-0910940 and by a grant from Yahoo! Research.

1 Introduction

Given a Markov chain M and a set U of objects, we seek an optimal assignment of objects to the states of the Markov chain. Consider a user walking through M viewing the objects at visited states. Associated with each object $u \in U$ is a probability p_u that the user exits the walk upon viewing u, together with a value ν_u that represents the user's utility on viewing u. Depending on the application, the Markov chain represents either the passage of a user's eyes over objects on a web page, or the transit of a user through pages on the Web. Our goal is to optimally place (some of) the objects on the states of M, with a maximum of one object per state. This problem arises in the algorithmic layout of Web pages and Web sites; we describe a series of (increasingly intricate) cases arising in practice:

(1) In classical web search, the objects are web pages; the search engine scores these for a query and places 10 top-scoring pages into the 10 slots on the search results page. The total benefit to the user of these 10 results is measured by the *rank-biased precision* [15], which is a geometrically weighted sum of the scores of the top 10 results. This measure can be viewed as the expected total score of pages seen by a user transiting through a Markov chain with 10 states, where at each step the user chooses to read a page (thereby obtaining utility) or not, and chooses to exit or not. If the probabilities of exiting are independent of the slot/object, the trivial strategy is to place the ten top-scoring documents in the 10 slots, in decreasing order of scores. More generally, these probabilities are object-dependent, making the problem harder (more on this below).

(2) In image search, the objects are images. The image search engine scores these for a query and must place top-scoring images into the slots in a rectangular matrix that is presented to the user. The standard algorithm sorts the images in decreasing order of score, and places them in row-major order in the matrix. However, eye-tracking studies show that users do not scan an image results page in row-major order — rather, their eyes tend to traverse the page in a cross-like trajectory, with some randomness [16]; see also Figure 2. We may view such user eye-tracks as a Markov chain with a state for each matrix position (thus the graph underlying M is a grid graph), with transition probabilities dominantly in the rough shape of a cross. In fact, such "non-linear" eye traversals are common even in page layouts other than the rectangular matrix [8, 2, 5]. Thus our formulation with Markov chains in two dimensions models settings such as Google's Universal search, Microsoft's Bing, and Yahoo's slotted direct displays, as well as many product and fare searches — in all cases, results pages have a two-dimensional placement of objects that are not only linear lists of links, but include photos, maps, fares etc. juxtaposed in two dimensions.

(3) A website consists of a set of web pages, with hyperlinks amongst them; a content publisher such as Yahoo! or CNN is a good model to keep in mind. The content on these pages induces a user to walk through the links following a Markov chain. The objects are advertisements, each of which has a payoff to the publisher (because the user views or clicks on the advertisement, with an associated payoff). The objective is to place the advertisements so as to maximize the publisher's total payoff. In this setting, the Markov chain does not have to be planar or have any simple structure.

1.1 Our contributions

The model. Let M be a Markov chain over a state space S of size n, where $M_{i,j}$ is the transition probability from state i to state j. Each state in S represents a "slot" that can be filled with an object (e.g., an image, an advertisement, a URL, a text snippet). The Markov chain also has two distinguished states s (*source*) and t (*sink*) that represent the beginning and the end states. The sink will not be assigned any object and $M_{t,t} = 1$.

Let U be a universe of objects. Associated with each *object* $u \in U$ is a pair $\langle p_u, \nu_u \rangle$, where $p_u \in [0, 1]$ is the *stopping probability*, i.e., the probability that the user stops after looking at object u and ν_u is the *utility* accrued if the user looks at object u.[1]

Let $\pi : S \to U \cup \{\perp\}$ be a mapping from states to objects, where $\perp$ means no object is placed in the state; by convention, $\pi(t) = \perp$. Let $\nu_\perp = p_\perp = 0$. Given such a mapping, the stochastic process works as follows. The user starts from the source s of the Markov chain M and performs a random walk according to M. If the user is currently in state i, a utility of $\nu_{\pi(i)}$ is accrued. The user then flips a coin and with probability $p_{\pi(i)}$ walks to t and with probability $1 - p_{\pi(i)}$ walks according to $M_{i,\star}$. Notice that the walk effectively ends and the total utility is frozen whenever the user reaches the sink t.

[1]We assume that all the probabilities are rationals, represented as a ratio of two $O(\log n)$-bit numbers. Also, in practice, we will have $|U| \gg n$; our approximation factors will be in terms of n.

Definition 1 (Markov Layout problem (MLP)). *Given a Markov chain M and a universe U of objects, the* Markov layout problem *(MLP) is to find an assignment π from the states S of M to $U \cup \{\perp\}$ such that the expected total utility is maximized and no object is assigned to more than one state.*

We focus on the following cases of MLP, depending on the structure of the input and the requirements on the output. These cases are directly motivated by our primary application domains; from a theoretical standpoint they also provide an understanding of the boundary cases of the hardness of the problem. We outline these cases through restrictions on the input structure, as well as on the output.

(1) *Input structure*: We consider settings where the graph structure of the underlying Markov chain can be exploited. The graph can be arbitrary or can be a DAG, a directed acyclic planar grid, tree, etc. Note that grid-like graphs are of particular interest for planar object layout on a web page; for example, this captures the layout of image search results or the layout of products on shopping webpages. We also address the special case where the stopping probability is the same for all the objects.

(2) *Output requirements*: In certain settings, we demand that the assignment π leaves no state (except t) unfilled, i.e., $\pi : S \setminus \{t\} \to U$; this is the *gap-free MLP*. This induces subtle but important differences both in our algorithms (as will be evident) and in practice. Leaving a gap in the assignment is equivalent to not distracting the user with an unnecessary advertisement or object, in pursuit of greater utility elsewhere in the layout; this phenomenon is in fact understood by content publishers and portals. In other settings, we relax the condition that the assignment is injective, i.e., we allow objects to be reused; this is the *multi-use MLP* (Appendix C).

Main results. Our main result is that MLP can be approximated to within a factor $O(\log n)$. The core idea behind the algorithm is to transform an optimization problem over partial permutations into an optimization problem over sets by losing a logarithmic factor in approximation; in the course of this, we define a linear version of MLP. We then show how to approximate this linear version by expressing it as a submodular function with two matroid constraints. As far as we know, this is the first time that an optimization problem over partial permutations, whose objective function is an unbounded-degree polynomial, is solved through a reduction to submodular maximization over sets. On the other hand, we show that MLP is APX-hard via a reduction from MAX-BISECTION.

In contrast, gap-free MLP turns out to be much harder. Even if the underlying graph is a DAG with a single self-loop, each object has unit utility, and the stopping probabilities are binary, we show that gap-free MLP is inapproximable to within $\Omega(2^{n^{1-\epsilon}})$; we also show this to be near-optimal. This result can be found in Section 5.

We then focus on important cases when the digraph underlying M has a special structure. We obtain an $O(\log^{3/2} n)$-approximation algorithm for the harder gap-free MLP on grids (Section 4) and an $O(\log n)$-approximation for gap-free MLP on directed trees (Appendix A); these algorithms work by carefully exploiting the geometry of the setting. We also obtain a quasi-PTAS for directed trees, generalizing the work of [13]. These special cases are appropriate for the applications that motivated this work.

A summary of the main results is shown in Table 1. All missing proofs are presented in the Appendix.

Remarks on the model. First, in our model, the utility accumulates as the states are revisited, whereas it is sometimes desirable to discount revisits to a state (in the extreme case, only the first visit to a state would be considered). Without becoming non-Markovian, we can model such a discounting loosely by reducing every transition probability by some factor.

Second, it is tempting to question the need for having both utility and stopping probability for an object. However, they are two different facets of an object: for example, in web search, if the query is information-seeking, search results with high utility do not necessarily have high stopping probability.

Third, one might consider the seemingly more general model where an object u is associated with a non-negative quintuple $\langle q_{u,1}, q_{u,2}, q_{u,3}, q_{u,4}, g_u \rangle$ with $q_{u,1} + q_{u,2} + q_{u,3} + q_{u,4} = 1$, where g_u is the utility if the user clicks on the object. A user looks at u and clicks and stops with probability $q_{u,1}$, does not click but stops with probability $q_{u,2}$, does not click and moves to the next state with probability $q_{u,3}$, and clicks and moves to the next state with probability $q_{u,4}$. By letting $\nu_u = g_u \cdot (q_{u,1} + q_{u,4})$ and $p_u = q_{u,1} + q_{u,2}$, it is easy to see that the expected total utility in this setting is the same as the original setting.

1.2 Related work

Aggarwal et al. [1] as well as Kempe and Mahdian [13] study special cases of the *linear cascade* model of Example (1) (see Section 1), in the context of sponsored search advertisements. For the special case when M consists of a line with all transition probabilities being equal, they give an exact solution to our problem. The authors of [13] also consider a more general model where the Markov chain's probabilities are themselves sampled from a distribution before the user navigates its slots — the aim is to optimize the ad placement given the distribution. For the general setting of Example (1), they give a 4-approximation algorithm, as well as a quasi-PTAS. Giotis and Karlin [10], Deng and Yu [7], and Gomes, Immorlica, and Markakis [11] study the equilibria of ad slot auctions using the model of Example 1. Craswell et al. [6] give some empirical evidence (from click logs) in support of the linear model; thus our extension for image search in two dimensions has a natural basis in their work. Charikar et al. [4] consider a seemingly related but in fact different problem: they have multiple Markov chains representing the behavior of two or more classes of users, each with its own Markov chain on a common set of states. They focus on inferring the user's class from a prefix of the user's trajectory; this information is used to place class-specific advertisements on the states. In [5] we report on the empirical study of several simple heuristics for the web image search problem, demonstrating tangible improvements in practice over the simple row-major ordering used hitherto in image search engines (Figure 3).

Problem	Approximation	Hardness
General MLP	$O(\log n)$ (Theorem 17)	APX-hard (Theorem 21)
Gap-free MLP ($b = n^{1+\epsilon}$)	$2^{O(b)}$ (Theorem 4)	$2^{n^{1-\epsilon}}$ (Theorem 22)
Gap-free MLP (Acyclic grids)	$O(\log^{3/2} n)$ (Theorem 20)	
Gap-free MLP (Directed trees)	$O(\log n)$ (Theorem 23)	

Table 1. Summary of results.

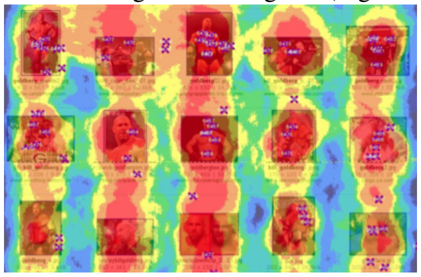
Figure 2. Eye-gaze on image search.

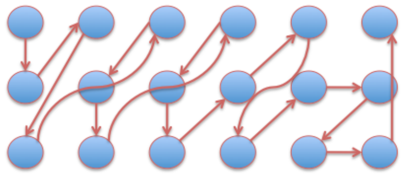
Figure 3. Inferred gaze ordering [5].

2 Warm-up: Easy cases

We begin with simple observations about the problem on general Markov chains.

Theorem 2. *MLP and gap-free MLP are in P if the optimum has infinite expected utility.*

Next, we address another important special case: when all the objects have the same stopping probability and we require that the solution be gap-free.

Theorem 3. *Gap-free MLP is in P if the stopping probability is the same for each object.*

Utilizing the characterization in Theorem 2, we can also show

Theorem 4. *Gap-free MLP can be approximated to within factor $2^{O(b)}$, if b is the number of bits in the input instance.*

Later, in Section 5, we show that the above approximation is essentially tight. Finally, we observe a trivial approximation for DAGs. The *depth* of the DAG is the length of the longest path from the source s.

Proposition 5. *MLP and gap-free MLP on DAGs can be approximated to within a factor d, where d is the depth of the DAG.*

3 An $O(\log n)$-approximation algorithm

In this section we present an $O(\log n)$-approximation algorithm for MLP. The idea behind the algorithm and the analysis is to reduce the objective function, which is an unbounded degree polynomial to be optimized over a set of partial permutations, to a more manageable submodular function over sets.

Here is an overview of the algorithm. To go from permutations to sets, we first do a series of simplifying transformations that only lose constant factors in approximations (Section 3.1). The next step is to create logarithmically many buckets with objects of similar ratios of utility to stopping probability, and focus on one such bucket; this will only cost a logarithmic factor in the approximation. In order to show that similar ratios are indeed the right level of granularity, we introduce a new stochastic process that replaces each high-utility object with a larger stopping probability by a sequence of smaller objects such that in expectation, the utility and the stopping probabilities are comparable (Section 3.2). This leads to a linear version of MLP. Finally, we show that the linear version is equivalent to a submodular function with two matroid constraints and hence is approximable to within a constant factor (Section 3.3).

3.1 Simplifying steps

We begin by considering the case when the stopping probabilities are zero for all the objects.

Lemma 6. *MLP can be solved optimally if $p_u = 0$ for all $u \in U$.*

Next, we show that it suffices to consider only the objects with positive stopping probabilities bounded away from 1 at the expense of losing a constant factor in the approximation.

Lemma 7. *Let $\gamma \in (0,1]$. If instances of MLP in which $\forall u \in U$, $p_u \in (0,\gamma)$ can be approximated to within a factor of α, then the general MLP can be approximated to within a factor $1+\alpha+1/\gamma$.*

Proof. Let A be a given instance of the Markov layout problem. Let $Z = 1+\alpha+1/\gamma$.

Now, consider an optimal solution π^* for A. The total value of π^* is the sum over the states x_i of the Markov chain of the utility of the object in x_i, $\nu_\pi(x_i)$, times the expected number of visits to x_i. Removing an object from an arbitrary state x_i voids that state's contribution to the sum, but does not decrease the contribution of other states to the sum. (In fact, leaving an empty state entails a stopping probability 0 in that state, so that the expected number of visits to each of the other states does not decrease.)

(1) Suppose the total utility of the states containing objects with zero stopping probability is at least $a = 1/Z$ of the total utility of π^*. Then, if we take the instance A, remove all the objects with positive stopping probability to obtain an instance B', and apply the SORT-AND-MATCH algorithm on B', we are guaranteed that the optimal solution to B' is a Z-approximation to the optimal solution to A.

(2) Suppose instead that the total utility of states containing objects with stopping probability at least γ is at least $b = 1/(\gamma Z)$ of the total utility of π^*. Then, if we remove all the objects with stopping probability at most γ from π^*, obtaining an assignment π'', we have that the total utility of π'' is at least $1/(\gamma Z)$ times the utility of π^*. On the other hand, since all the objects in π'' have continuing probability at most $1-\gamma$, we have that, in expectation, we will visit at most $\sum_{i=0}^{\infty}(1-\gamma)^i = 1/\gamma$ objects in a walk. The utility of π'' is therefore at most $1/\gamma$ times the maximum utility of an object in π''. Therefore, taking the object with stopping probability at most γ and with the largest utility, and placing it in the source state s, gives a $1/\gamma$-approximation to the utility of π'', and hence a Z-approximation to the utility of π^*.

(3) Finally, suppose that at least $c = \alpha/Z$ of the total utility π^* is given by objects in A with stopping probability in $(0,\gamma)$. We then remove from A all the objects with stopping probability not in $(0,\gamma)$ to obtain an instance B. An α-approximate solution to B will then be a Z-approximation to A.

Since $a+b+c=1$, at least one of the premises of three cases holds, and the claim follows. □

Let $p_{\min}$ be the minimum non-zero stopping probability in the given instance. Since stopping probabilities are ratios of non-negative integers of $O(\log n)$ bits each, we have $p_{\min} = \min_{\substack{u\in U\\ p_u>0}} p_u \geq \frac{1}{\mathrm{poly}(n)}$.

Next, we show that once again by sacrificing a small factor in approximation, we can assume that the maximum ratio between two object utilities is $1/(\epsilon p_{\min})$, for every sufficiently small constant ϵ.

Lemma 8. *For any $0 < \epsilon < 1$, if instances of MLP in which $\forall u \in U, p_u \in (0,\gamma)$ and the ratio of the utilities of two distinct objects are in $\left[1, \frac{1}{\epsilon p_{\min}}\right]$ can be approximated to within a factor α, then instances of MLP where all the stopping probabilities are in $(0,\gamma)$, can be approximated to within a factor $(1+\epsilon)\alpha$.*

Proof. Let u be the most valuable object, i.e., $u = \text{argmax}_{u \in U} \nu_u$ in the given instance A. We claim that removing from the instance all objects $u' \neq u$ such that $\nu_{u'} \leq \epsilon p_{\min} \nu_u$ still guarantees that the resulting instance B has an optimal solution within a $(1+\epsilon)$ factor of the optimal solution to A.

Indeed, since there is no object in A with zero stopping probability, we have $p_{\min} = \min_{u \in U} p_u > 0$, and thus an upper bound on the expected number of objects that can be visited in the random walk is

$$\sum_{i=0}^{\infty} (1 - p_{\min})^i = \frac{1}{p_{\min}}.$$

Removing from the solution all the objects u' such that $\nu_{u'} \leq \epsilon p_{\min} \nu_u$ to obtain B reduces the solution's utility by at most the expected number of such u''s we can encounter in a random walk times their maximum utility, i.e., $(1/p_{\min})\epsilon p_{\min} \nu_u = \epsilon \nu_u$. Finally, observing that a solution that places u in the starting state s has utility at least ν_u yields the claim that a α-approximate solution to B is a $((1 \pm \epsilon)\alpha)$-approximate solution to A. Note that since the smallest utility is positive, rescaling the utilities so that the minimum is 1 does not change the approximation ratio of the problem. □

3.2 $\bar{k}$-process and linear MLP

We will now consider a generalization of our process. This will be useful in the next reduction for grouping together objects with (very) different utilities and stopping probabilities. The generalization shows that our process — in which, the user visits a state, a utility is accrued, and *then* possibly a stop event is triggered — can be well-approximated by a continuous process where the user visits a state and an arbitrarily long sequence of gains/possible-stops happens. The generalization is parametrized by $\bar{k} = (k_{u_1}, k_{u_2}, \ldots)$ where, $\forall u \in U$, k_u is a positive integer.

Definition 9 ($\bar{k}$-process)**.** *When the user sees the object u, a process of k_u rounds is started: at the beginning of each round, a utility of ν_u / k_u is gained. Then, a coin with head probability $1 - \sqrt[k_u]{1-p_u}$ is flipped independently of previous flips. If heads comes up, then the user* negatively stops *the process. Otherwise, the user moves on to the next round, or the user* positively stops *the process if all k_u rounds have been played.*

Clearly, our original Markov process is a $\bar{1}$-process: a negative stop corresponds to a stopping event and a positive stop corresponds to a new move according to the Markov chain. We now prove a lemma that compares the utilities accrued by the $\bar{k}$-processes and our original process, when the user sees an arbitrary object u. Let $\mu_u(k_u)$ be the expected utility accrued when the user sees object u according to the $\bar{k}$-process.

Lemma 10. *For each $k_u \geq 1$, we have $\nu_u \geq \mu_u(k_u) \geq f(p_u)\nu_u$, where $f(p) = p(\ln \frac{1}{1-p})^{-1}$ for $p \in (0,1)$ and $f(p) = 1 - p$ for $p = 0, 1$. Furthermore, $f(p)$ is a decreasing function of p, for $p \in [0,1]$. Also, the probability that the user negatively stops the process while looking at u is p_u for each $k_u \geq 1$.*

Proof. First of all, observe that the lower bound on the expected utility of the $\bar{k}$-process at state x_i is trivial if $p_u = 0, 1$. In the former case, a gain of ν_u / k_u will be accrued at each of the k_u steps spent in state x_i, so the utility will be ν_u with probability 1. In the latter case, the lower bound equals 0, so the statement follows from the observation that no utility can be negative. Therefore, we assume $p_u \in (0,1)$.

By definition, $\mu_u(1) = \nu_u$. Now, the expected utility of object u in the $\bar{k}$-process is

$$\mu_u(k_u) = \frac{\nu_u}{k_u} \sum_{j=0}^{k_u - 1} \left(\sqrt[k_u]{1-p_u}\right)^j = \nu_u \frac{p_u}{k_u\left(1 - \sqrt[k_u]{1-p_u}\right)}.$$

We upper bound the latter denominator so to get a lower bound for $\mu_u(k_u)$. Recall that, for each $x \in (0,1)$, it holds that $(1-x)^\alpha = \sum_{n=0}^{\infty} \binom{\alpha}{n}(-x)^n$, where $\binom{\alpha}{n}$ is the generalized binomial $\binom{\alpha}{n} = \frac{\alpha^{\underline{n}}}{n!}$. Therefore,

$$\begin{aligned} k_u\left(1-\sqrt[k_u]{1-p_u}\right) &= -k_u \sum_{j=1}^{\infty}\left(\frac{(1/k_u)^{\underline{j}}}{j!}(-p_u)^j\right) \\ &= -k_u \sum_{j=1}^{\infty}\left(\frac{1/k_u\,(1/k_u-1)^{\underline{j-1}}}{j!}(-p_u)^j\right) = \sum_{j=1}^{\infty}\left(\frac{(1-1/k_u)^{\overline{j-1}}}{j!}p_u^j\right) \\ &\leq \sum_{j=1}^{\infty}\left(\frac{(j-1)!}{j!}p_u^j\right) = \sum_{j=1}^{\infty}\left(\frac{1}{j}p_u^j\right) = \ln\frac{1}{1-p_u}, \end{aligned}$$

where the last step follows from the well-known identity $\ln(1-x) = -\sum_{n=1}^{\infty}\frac{x^n}{n}$, which holds for $x \in (-1,1)$. The upper bound on $\mu_u(k_u)$ is trivial since the maximum utility one can accrue during a visit of x_i in the $\overline{k}$-process is $k_u \times \nu_u/k_u$.

Next, observe that

$$f(x) = \frac{x}{\ln\frac{1}{1-x}} = \frac{x}{\sum_{j=1}^{\infty}\left(\frac{1}{j}x^j\right)} = \frac{1}{1+\sum_{j=1}^{\infty}\left(\frac{1}{j+1}x^j\right)}.$$

Finally, since $\lim_{x\to 0^+} f(x) = 1$ and $\lim_{x\to 1^-} f(x) = 0$, and since $f(x) \in (0,1)$ for each $x \in (0,1)$, we conclude that $f(x)$ is decreasing in $[0,1]$.

Clearly, the probability that the user negatively stops the process is p_u. Indeed, the probability of a negative stop event is $1 - \left(\sqrt[k_u]{1-p_u}\right)^{k_u} = p_u$. □

For each object $u \in U$ with $p_u > 0$, we associate the integer $k_u = \lceil p_u/p_{\min}\rceil$. Given a path Ψ in the Markov chain, $\Psi = (x_1, \ldots, x_{|\Psi|})$, and an assignment of objects π to the states of the Markov chain, we let $\Psi^+(i) = \Psi^+_\pi(i)$ be the object in the ith non-empty state in Ψ with the assignment π, and $S_\pi(\Psi)$ be the number of non-empty states in Ψ. We also let $\mathcal{P} = \mathcal{P}_\pi$ be the set of directed paths Ψ starting at the source s, and ending in the sink t, that pass through at least one non-empty state. Finally, let $K = K_{\Psi,\pi}$ be the sum of the k_u's of the objects u in path Ψ in the assignment π, $K_{\Psi,\pi} = \sum_{j=1}^{S_\pi(\Psi)} k_{\Psi^+(j)}$.

We now define a new problem and show that its solutions can be used to approximate the original MLP.

Definition 11 (Linear MLP). *Given an assignment π, we define*

$$V_L(\pi) = \sum_{\Psi\in\mathcal{P}}\left(\min\left(K_{\Psi,\pi}, \left\lceil\frac{1}{p_{\min}}\right\rceil\right)\prod_{j=1}^{|\Psi|-1} M_{x_j,x_{j+1}}\right).$$

The linear MLP *is to maximize $V_L(\pi)$ subject to the same conditions as in the general MLP.*

By the same conditions, we mean that no object can be used more than once and each state can be assigned at most one object. We now show that, under some assumptions, an approximate solution to the linear MLP is an approximate solution to MLP.

Lemma 12. *Suppose that an instance of MLP is such that all the stopping probabilities are in $(0, 1/2)$ and all the utilities are in $[1, 1/(\epsilon p_{\min})]$. Then there exists some $0 \leq i < \lceil\log 1/(\epsilon p_{\min}^2)\rceil = T$, such that if we remove from the instance all the objects u for which $\nu_u/p_u \notin [2^i, 2^{i+1})$, then if an assignment of the remaining objects is a c-approximation to the linear MLP, then it is also an $O(cT)$-approximation to the original instance.*

Proof. Let $V(\pi)$ be the expected utility of the assignment π on the given instance of MLP. Then[2],

$$V(\pi) = \sum_{\Psi=(s=x_0,\ldots,x_{|\Psi|-1})}\left(\prod_{j=1}^{|\Psi|-1}\left((1-p_{\pi(x_{j-1})})M_{x_{j-1},x_j}\right)p_{\pi(x_{|\Psi|-1})}\sum_{i=0}^{|\Psi|-1}\nu_{\pi(x_i)}\right).$$

[2]Recall that if t is the sink, then $\nu_{\pi(t)} = 0, p_{\pi(t)} = 1$.

We now partition the objects in U into buckets, where the bucket B_i will contain all objects $u \in U$ such that $\nu_u/p_u \in \left[2^i, 2^{i+1}\right)$ with $0 \le i < \left\lceil \log \frac{1}{\epsilon p_{\min}^2} \right\rceil = T$. Observe that $T = O(\log n)$.

Consider an optimal solution V^* (achieved by π^*) to the given instance of MLP. We can rewrite its utility as

$$V^* = V(\pi^*) = \sum_{x \neq t} \left(\nu_{\pi^*(x)} E[\text{\# of times } x \text{ is reached}]\right)$$
$$= \sum_{i=0}^{T-1} \sum_{\substack{x \neq t \\ \pi^*(x) \in B_i}} \left(\nu_{\pi^*(x)} E[\text{\# of times } x \text{ is reached}]\right).$$

Let i^* be a (not necessarily unique) index of the outer sum of the previous expression that maximizes the inner sum. Observe that the inner sum for $i = i^*$ is the utility obtained by considering only the utilities of the objects in bucket B_{i^*}. Suppose we remove all other objects. The expected number of visits to a state, for each state, is not decreased (since an *empty* state, i.e., a state with a gap, has stopping probability zero). Thus, by our optimal choice of B_{i^*}, we have that the utility of the new solution is at least $\frac{V^*}{T}$.

For each $0 \le i < T$, we create the instance D_i containing the objects in B_i. We are guaranteed that at least for one of the D_i's, its best solution will be a T-approximation to the given instance. For the generic instance $D = D_i$, whose every object u is such that $\rho = 2^i \le \nu_u/p_u < 2^{i+1} = 2\rho$, we invoke Lemma 10.

Since $p_u \le 1/2$ by our assumptions, Lemma 10 guarantees that the value of any solution to MLP is within a factor of $1/f(1/2) = \ln 4$ of the value of the same solution to the $\bar{k}$-process problem. Therefore, if we c-approximate the $\bar{k}$-process MLP, then we obtain a $(cT \ln 4)$-approximation to our original MLP. We now show that the objective function of the $\bar{k}$-process MLP approximates $V_L(\cdot)$, the objective function of linear MLP.

We start by analyzing the per-round stopping probability $\bar{p}_u$ of the $\bar{k}$-process instance, for the generic object u.

$$\begin{aligned} \bar{p}_u &= 1 - \sqrt[k_u]{1-p_u} = 1 - (1-p_u)^{\left\lceil \frac{p_u}{p_{\min}} \right\rceil^{-1}} \ge 1 - (1-p_u)^{\left(\frac{p_u}{p_{\min}}+1\right)^{-1}} \\ &\ge 1 - (1-p_u)^{\frac{p_{\min}}{2p_u}} \ge 1 - e^{-\frac{p_{\min}}{2}} \ge \frac{p_{\min}}{2} - \frac{1}{2}\left(\frac{p_{\min}}{2}\right)^2 \ge \frac{7}{16} p_{\min}, \end{aligned}$$

since $p_{\min} \le \frac{1}{2}$. Furthermore,

$$\bar{p}_u = 1 - (1-p_u)^{\left\lceil \frac{p_u}{p_{\min}} \right\rceil^{-1}} \le 1 - (1-p_u)^{\frac{p_{\min}}{p_u}} \le 1 - \left(\frac{1}{4}\right)^{p_{\min}} \le (2\ln 2) p_{\min},$$

where the middle inequality follows from $(1-x)^{x^{-1}} \ge 1/4$, for each $x \in [0, 1/2]$. On the other hand, the per-round gain $\bar{\nu}_u$ can be bounded by

$$\frac{\rho}{2} p_{\min} \le \nu_u \frac{p_{\min}}{2p_u} \le \nu_u \frac{p_{\min}}{p_u + p_{\min}} \le \bar{\nu}_u \le \nu_u \frac{p_{\min}}{p_u} \le 2\rho p_{\min}.$$

Therefore, for each object u in instance D in the $\bar{k}$-process, u's per-round stopping probability $\bar{p}_u \in [(7/16)p_{\min}, (2\ln 2)p_{\min}]$ and u's per-round gain $\bar{\nu}_u \in [(1/2)\rho, 2\rho]$.

Consider the following process, which is equivalent to our $\bar{k}$-process: the user chooses a path $\Psi \in \mathcal{P}$ according to its probability given by the Markov chain, i.e., independent of the objects' placement. Then, each time the user gets to an object u in the path, the user runs the $\bar{k}$-process (i.e., for at most k_u rounds, a gain of $\bar{\nu}_u$ will be accrued at the beginning of each round, and the stopping event at the end of a round will happen with probability $\bar{p}_u$.) The user will follow the path until it ends at t, independent of the stopping events. Only, when a stopping event happens, the user will not get the utilities of the remaining rounds in the path.

Let $g_\pi(\Psi)$ be the expected utility of path Ψ, conditioned on path Ψ to be followed by the user and let $V_D(\pi)$ be the utility of the instance D with the $\bar{k}$-process. Then,

$$V_D(\pi) = \sum_{\Psi \in \mathcal{P}} \left(g_\pi(\Psi) \prod_{j=1}^{|\Psi|-1} M_{x_j, x_{j+1}} \right),$$

$$g_\pi(\Psi) = \sum_{i=1}^{S_\pi(\Psi)} \left(\bar{\nu}_{\Psi^+(i)} \prod_{j=1}^{i-1} \left(1 - \bar{p}_{\Psi^+(j)}\right)^{k_{\Psi^+(j)}} \sum_{j=0}^{k_{\Psi^+(i)}-1} \left(1 - \bar{p}_{\Psi^+(i)}\right)^j \right).$$

Let $K = K_{\Psi,\pi} = \sum_{j=1}^{S_\pi(\Psi)} k_{\Psi^+(j)}$. We will show an upper and lower bound on $g_\pi(\Psi)$ in terms of K.

Lemma 13. $\rho\frac{(1-\ln 2)^2}{2}\min\left(K, \left\lceil\frac{1}{p_{\min}}\right\rceil\right) \le g_\pi(\Psi)$ *and* $g_\pi(\Psi) \le \rho\frac{32}{7}\min\left(K, \left\lceil\frac{1}{p_{\min}}\right\rceil\right)$.

From Lemma 13, we obtain $\frac{7}{32}V_D(\pi) \le \rho V_L(\pi) \le \frac{2}{(1-\ln 2)^2}V_D(\pi)$. Therefore a c-approximate solution to $V_L(\pi)$ is then a $\left(\frac{64}{7(1-\ln 2)^2}c\right)$-approximate solution to $V_D(\pi)$, which is then a $\left(\frac{64\ln 4}{7(1-\ln 2)^2}Tc\right)$-approximate solution to MLP. □

3.3 A constant-factor approximation for linear MLP

Next, we show that linear MLP can be approximated to within a constant factor in polynomial time.

Theorem 14. *Linear MLP is approximable to within factor $\frac{7}{2}+\epsilon$ in polynomial time.*

Proof. To show that $\max_\pi V_L(\pi)$ can be approximated in polynomial time, we prove that $V_L(\pi)$ is equivalent to a submodular function with two matroids constraints (Lemma 15) and for a given assignment π, the utility of $V_L(\pi)$ can be computed in polynomial time (Lemma 16). Given these, we can use the submodular optimization algorithm of Lee, Sviridenko, and Vondrák [14] under $k = 2$ matroid constraints to yield an approximation of $k+1+\frac{1}{k}+\epsilon = \frac{7}{2}+\epsilon$. □

Lemma 15. *$V_L(\pi)$ is equivalent to a submodular function with two matroid constraints.*

Proof. To prove submodularity, we first need to be more specific regarding the solution space of the $V_L(\pi)$ problem. First of all, to each object u in the instance we associate an integer k_u as before. Then, we consider the universe $U \times S$, where S is the set of states of the Markov chain. A solution to the problem is a subset of $U \times S$. Note that in general such a solution may not be feasible for $V_L(\pi)$ since an object might be used in more than one state or a state might be assigned more than one object. To ensure feasibility for $V_L(\pi)$, we add two matroid constraints: (i) the *object matroid* independent sets that contain, for each object u, at most one pair containing the object u and (ii) the *state matroid* independent sets that contain, for each state x, at most one pair containing the state x. These constraints ensure that an object is used at most once and a slot is filled at most once.

Observe that the intersection $\mathcal{I} = \mathcal{I}_o \cap \mathcal{I}_s$ of an independent set $\mathcal{I}_o$ of the object matroid and an independent set $\mathcal{I}_s$ of the state matroid can be easily transformed into a feasible solution $\pi = \pi(\mathcal{I})$ for $V_L(\pi)$ in polynomial time. Conversely, each feasible solution of $V_L(\pi)$ can be transformed in polynomial time into an independent set of the object and the state matroids. Indeed, the intersection is such that each state contains at most one object and each object is contained in at most one state, which is exactly the definition of feasible solution of $V_L(\pi)$.

For a path $\Psi = (x_1, \ldots, x_{|\Psi|}) \in \mathcal{P}$ and $A \subseteq U \times S$, let $K'_{\Psi,A} = \sum_{i=1}^{|\Psi|}\sum_{(u,x_i)\in A} k_u$. Observe that, if A satisfies both matroid constraints and $\pi = \pi(A)$ is the assignment induced by A, then $K'_{\Psi,A} = K_{\Psi,\pi(A)}$, i.e., $K' = K$ when A is a valid assignment for the linear MLP. The objective $V_S(A)$ with $A \subseteq U \times S$ is

$$V_S(A) = \sum_{\Psi\in\mathcal{P}}\left(\min\left(K'_{\Psi,A}, \left\lceil\frac{1}{p_{\min}}\right\rceil\right)\prod_{j=1}^{|\Psi|-1} M_{x_j,x_{j+1}}\right)$$

Note that $V_S(A) = V_L(\pi(A))$ for each A satisfying the two matroid constraints.

We now prove that $V_S(A)$ is a submodular function; observe that $V_S(A)$ is a weighted sum of terms $T_\Psi(A) = \min\left(K'_{\Psi,A}, \lceil 1/p_{\min}\rceil\right) = \min\left(\sum_{i=1}^{|\Psi|}\sum_{(u,x_i)\in A} k_u, \lceil 1/p_{\min}\rceil\right)$, one for each path Ψ. We show that each of the terms is submodular, thus showing the submodularity of $V_S(A)$. Given two arbitrary sets $A, B \subseteq U \times S$, the following holds.

(i) If $K'_{\Psi,A} > \lceil 1/p_{\min}\rceil$, then $K'_{\Psi,A\cup B} > \lceil 1/p_{\min}\rceil$, and $T_\Psi(A\cup B) = T_\Psi(A)$. Also, in general, $T_\Psi(A\cap B) \le T_\Psi(B)$. Thus, $T_\Psi(A\cup B) + T_\Psi(A\cap B) \le T_\Psi(A) + T_\Psi(B)$, and the function is submodular.

(ii) If $K'_{\Psi,B} > \lceil 1/p_{\min}\rceil$, a similar reasoning applies.

(iii) Otherwise we have $K'_{\Psi,A}, K'_{\Psi,B} \le \lceil 1/p_{\min}\rceil$ and in this case, $T_\Psi(A) = K'_{\Psi,A}$, $T_\Psi(B) = K'_{\Psi,B}$ and $T_\Psi(A\cap B) = K'_{\Psi,A\cap B}$. In general, $T_\Psi(A\cup B) \le K'_{\Psi,A\cup B} = K'_{\Psi,A} + K'_{\Psi,B} - K'_{\Psi,A\cap B}$. Thus, in our case, $T_\Psi(A\cup B) \le T_\Psi(A) + T_\Psi(B) - T_\Psi(A\cap B)$; the function is then submodular. □

Lemma 16. *$V_L(\pi)$ can be computed in polynomial time.*

Proof. Let S_A be the set of states of the Markov chain that appear in at least one pair in A. We call these states "non-empty". Given a non-empty state $y \in S_A$ and an arbitrary state x in the Markov chain, let $q_{x,y}(A)$ be the probability that a random walk starting in x hits y before hitting any other non-empty state. For each non-empty state x and arbitrary state y, $q_{x,y}(A)$ can be obtained by well-known techniques in polynomial time. Start by copying the state x, and its out-links with their probabilities[3], obtaining a new state x', which will be empty (even if x was not so). Then, remove each outgoing edge from each non-empty state, and add to each non-empty state a self-loop having probability 1. Finally, compute the probability of ending up in the recurrent set $\{y\}$ if starting from x'. This can be done with classical techniques in polynomial time, i.e., by inverting the fundamental matrix of the new Markov chain.

Observe that the expression for $V_S(A)$ sums, for each path in $\mathcal{P}$, its probability times the minimum of the sum of the k_u's of the objects u it hits and $\lceil 1/p_{\min} \rceil$. If we group paths in $\mathcal{P}$ according to the sum of their k_u's (either $K' = 1, \ldots, \lceil 1/p_{\min} \rceil - 1$, or $K' \geq \lceil 1/p_{\min} \rceil$) then we can obtain $V_S(A)$ by summing, for each $k \in \{1, \ldots, \lceil 1/p_{\min} \rceil - 1\}$, the product of k times the probability of following some path in the kth group (i.e., our expected utility from paths whose objects' k_u's sum up to k), and, finally, by adding the product of $\lceil 1/p_{\min} \rceil$ times the probability of following a path whose objects k_u's sum up to at least $\lceil 1/p_{\min} \rceil$. The probabilities in our sum can be obtained using the $q_{x,y}(\pi)$'s. If $S_k(A)$ is the set of sequences σ having the initial state as first element, followed by non-empty states, $\sigma \in \{s\} \times \bigcup_{i=0}^{\lceil 1/p_{\min} \rceil} (S_A)^i$, such that $\sum_{i=1}^{|\sigma|} \sum_{(u,\sigma(i)) \in A} k_u = k$, then the probability of following some path in the kth group, $1 \leq k < \lceil 1/p_{\min} \rceil$, is given by

$$P_k(A) = \sum_{\sigma \in S_k(A)} \left(\prod_{i=1}^{|\sigma|-1} q_{\sigma(i),\sigma(i+1)} \left(1 - \sum_{x \in S_A} q_{\sigma(|\sigma|),x} \right) \right)$$

and

$$P_{\geq \lceil 1/p_{\min} \rceil}(A) = \sum_{\sigma \in S_k(A)} \prod_{i=1}^{|\sigma|-1} q_{\sigma(i),\sigma(i+1)},$$

which is the probability of following some path in the last group. Observe how the $P_k(A)$'s, and $P_{\geq \lceil 1/p_{\min} \rceil}(A)$ can be easily computed via dynamic programming. We can then express $V_L(\pi)$ as

$$V_L(\pi) = \sum_{k=1}^{\lceil 1/p_{\min} \rceil - 1} (k P_k(A)) + \left\lceil \frac{1}{p_{\min}} \right\rceil P_{\geq \lceil 1/p_{\min} \rceil}(A).$$

Given the expressions, it is clear that $V_L(\pi)$ can be computed in polynomial time. □

Finally, combining Lemma 7, Lemma 8, Lemma 12, and Theorem 14 we obtain the main result.

Theorem 17. *MLP can be approximated to within a factor $O(\log n)$.*

By slightly modifying Lemma 12, we can also prove the following result:

Corollary 18. *If, in the input instance of MLP, there are objects $u_1, \ldots, u_t$ such that for every object u, (ν_u/p_u) is within a constant fraction of (ν_{u_i}/p_{u_i}) for some i, then MLP can be approximated to within a factor $O(t)$.*

Thus, if MLP is inapproximable to better than $\Omega(\log n)$, then it is so only for instances that have $t = \Omega(\log n)$ objects $u_1, \ldots, u_t$ such that $\nu_{u_1}/p_{u_1} \geq c\nu_{u_2}/p_{u_2} \geq \cdots \geq c^{t-1}\nu_{u_t}/p_{u_t}$, for some constant $c > 1$.

4 A gap-free $O(\log^{3/2} n)$-approximation for acyclic grids

The algorithm for MLP described in Section 3 can produce assignments with gaps, i.e., there can be states in the Markov chain to which no object is assigned. In this section, we consider the gap-free MLP for grid graphs and obtain an $O(\log^{3/2} n)$-approximation; note that Theorem 17 implies an $O(\log n)$-approximation but with gaps. As we discussed earlier, grid graphs arise when objects (say image search results) are to be laid out in a grid.

[3]Making sure that if x had a self-loop, that would be copied into a link going from x' to x.

Definition 19 (Grid graph)**.** *The* grid graph *of order n is a $(2n+1) \times (2n+1)$ grid, with nodes $s_{i,j}$, for $-n \leq i, j \leq n$ — node $s_{0,0}$ being the source node. Each node assigns the same probability to each of its out-neighbors. Node $s_{i,j}$ has edges to node (i) $s_{i+1,j}$ if $0 \leq i < n$, or to the sink t if $i = n$; (ii) $s_{i-1,j}$ if $0 \geq i > -n$, or to the sink t if $i = -n$; (iii) $s_{i,j+1}$ if $0 \leq j < n$, or to the sink t if $j = n$; and (iv) $s_{i,j-1}$ if $0 \geq j > -n$, or to the sink t if $j = -n$.*

The main idea is as follows. First of all, it suffices to work on any one of the four quadrants of the grid if we allow our algorithm to lose a factor of 4. Then, observe that if the walk actually ends up in this quadrant, it is likely to remain within a cone around the bisector of that quadrant, since, at each round, the probability of going a step away from the diagonal is equal to the probability of going a step closer to it. Thus, we can lower bound our utility by what we would obtain by filling the cone with objects having roughly the same stopping probability. To upper bound the value of the optimal solution, we observe that the walk, regardless of how objects are placed, is nearly uniformly mixed inside a slightly smaller cone. Since the ratio between the size of the cones happens to be only polylogarithmic, we obtain our approximation ratio.

Theorem 20. *Gap-free MLP is approximable to within a factor $O(\log^{3/2} n)$ on any grid graph of order n.*

5 Hardness results

In this section we show hardness results for MLP and gap-free MLP in fairly restrictive settings; the latter result shines a spotlight on how algorithmically hard it is to require each state to be filled. We begin by showing that MLP is APX-hard.

Theorem 21. *MLP is APX-hard even if the stopping probabilities and the utilities are in $\{0, 1\}$.*

Note that from Corollary 18, instances of MLP satisfying property of the previous reduction (and, more generally, instances in which the number of different types of objects is constant), can be solved approximately with ratio $O(1)$ in polynomial time. The corresponding restricted MLP is therefore APX-complete.

Next, we show that gap-free MLP is hard to approximate when the underlying graph is a DAG with just one self-loop and the stopping probabilities are binary.

Theorem 22. *It is NP-hard to approximate gap-free MLP to within $2^{n^{1-\epsilon}}$, for each $\epsilon > 0$, even if (i) the graph is a DAG with a single self-loop and (ii) $\forall u$, $\nu_u = 1$ and $p_u \in \{0, 1\}$.*

In the reduction, the transitions are uniform over the out-neighbors for all but one state. It is possible to eliminate this exception by increasing the number of cycles. It is even possible to let all states have zero stopping probability. Also, observe that in terms of bit-complexity b (i.e., the number of bits needed to represent the instance, i.e., $O(\log n)$ bits per edge, plus the bits needed to represent the various probabilities), Theorem 22's instances have $b = n^{1+O(\epsilon)}$. Therefore, the proof of Theorem 22 also guarantees an inapproximability bound of $2^{b^{1-\epsilon}}$. By Theorem 4, this factor is nearly tight.

Acknowledgments

Flavio Chierichetti thanks Michael Mitzenmacher for many fruitful discussions on the linear model of [1].

References

[1] G. Aggarwal, J. Feldman, S. Muthukrishnan, and M. Pal. Sponsored search auctions with Markovian users. In *Proc. 4th WINE*, pages 621–628, 2008.

[2] A. Aula and K. Rodden. Eye-tracking studies: More than meets the eye, 2009. `http://googleblog.blogspot.com/2009/02/eye-tracking-studies-more-than-meets.html`.

[3] P. Berman and M. Karpinski. On some tighter inapproximability results (extended abstract). In *Proceedings of the 26th International Colloquium on Automata, Languages and Programming*, 1999.

[4] M. Charikar, R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. On targeting Markov segments. In *Proc. 31st STOC*, pages 99–108, 1999.

[5] F. Chierichetti, R. Kumar, and P. Raghavan. Optimizing two-dimensional search results presentation. In *Proc. 4th WSDM*, 2011.

[6] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proc. 1st WSDM*, pages 87–94, 2008.

[7] X. Deng and J. Yu. A new ranking scheme of the GSP mechanism with Markovian users. In *Proc. 5th WINE 2009*, pages 583–590, 2009.

[8] A. Duchowski. A breadth-first survey of eye tracking applications. *Behavior Research Methods, Instruments, and Computers*, 34(4), 2002.

[9] U. Feige, M. Karpinski, and M. Langberg. A note on approximating max-bisection on regular graphs. *Information Processing Letters*, 79(4):181 – 188, 2001.

[10] I. Giotis and A. Karlin. On the equilibria and efficiency of the GSP mechanism in keyword auctions with externalities. In *Proc. 4th WINE*, pages 629–638, 2008.

[11] R. Gomes, N. Immorlica, and E. Markakis. Externalities in keyword auctions: An empirical and theoretical assessment. In *Proc. 5th WINE*, pages 172–183, 2009.

[12] G. Gutin and A. Yeo. Note on maximal bisection above tight lower bound. *Information Processing Letters*, 110(21):966 – 969, 2010.

[13] D. Kempe and M. Mahdian. A cascade model for externalities in sponsored search. In *Proc. 4th WINE*, pages 585–596, 2008.

[14] J. Lee, M. Sviridenko, and J. Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. In *Proc. 12th APPROX*, pages 244–257, 2009.

[15] A. Moffat and J. Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM TOIS*, 27(1), 2008.

[16] S. Outing and L. Ruel. The best of eyetrack III: What we saw when we looked through their eyes. `http://www.poynterextra.org/eyetrack2004/main.htm`.

A Improved approximations for directed rooted trees

In this section we consider directed trees and obtain two improvements: (i) a gap-free $O(\log n)$-approximation algorithm and (ii) a gap-free quasi-polynomial time approximation scheme. Note that rooted trees are similar to the F-like eye-tracks on a page reported by [2] and to the placement of advertisements in tree-like websites. Therefore, they are interesting objects to study in their own right. First, we show that both the general and gap-free MLP on the directed tree can be approximated to within factor $O(\log n)$.

Theorem 23. *Gap-free MLP is approximable to within a factor $O(\log n)$ on any directed tree of size n.*

Next, we obtain a gap-free $(1 + \epsilon)$-approximation algorithm for MLP on directed rooted trees. This algorithm, however, runs in time quasi-polynomial in the tree size. This algorithm is similar in spirit to the Kempe–Mahdian's quasi-PTAS for lines [13]. Since lines are trees, this algorithm can be seen as a generalization of the quasi-PTAS of [13].

Theorem 24. *For any $\epsilon > 0$, gap-free MLP can be approximated to within a factor $(1+\epsilon)$ on any rooted directed tree of size n. The algorithm runs in time $n^{O(\log^2 n)}|U|$.*

We also note that the same dynamic programming approach can be used to obtain a quasi-polynomial time algorithm for bounded-treewidth graphs. We omit the details in this version.

B General vs. gap-free solutions

We show that the ratio between the general and the gap-free solutions of the Markov layout problem is unbounded. Consider the graph depicted in Figure B, where edges are directed from left to right. The transition probabilities are uniform over the set of out-neighbors.

Suppose we are given two "good" objects with utility 1 and stopping probability 0, and $n - 2$ "bad" objects with utility 0 and stopping probability 1. Then, the best solution for the general Markov layout problem is 2 (place the two good objects at s and x, and leave other slots empty). On the other hand, the best solution for the gap-free Markov layout problem is $1 + \frac{1}{n-2}$ (place a good object at s, a bad object at x, and fill the remaining states with the remaining $n - 3$ bad objects, together with the second good object). This gadget can be easily extended to show an unbounded gap between the two problems.

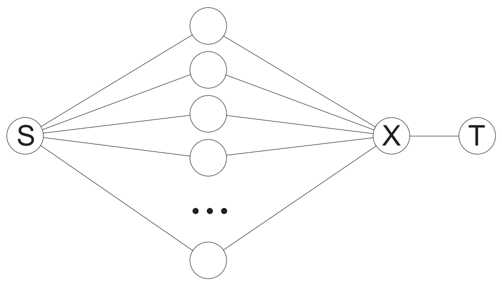

Figure 1: An example for the general vs. gap-free solutions.

C Multi-use Markov layout

Here, we show a simple result on the multi-use version of the Markov layout problem. The proof is omitted in this version.

Theorem 25. *The multi-use Markov layout problem can be solved in polynomial time if (i) the input graph is a DAG (possibly with self-loops), or if it can be made a DAG (possibly with self-loops) with the removal of constantly many nodes or (ii) the probability of transitioning to the sink from each other state is the same, or (iii) each object has the same expected utility.*

D Proofs for Section 2

D.1 Proof of Theorem 2

Proof. First, observe that a general instance of the Markov layout problem (possibly with gaps) can be transformed into a gap-free instance by adding n new objects to the universe, each with zero stopping probability and zero utility. We thus focus on the gap-free version.

Let $U_0 = \{u \in U \mid p_u = 0\}$ and $U_1 = \{u \in U \mid p_u = 1\}$. We claim that a solution with infinite utility exists if and only if the following hold:

(1) $\exists u^* \in U_0$ such that $\nu_{u^*} > 0$, and

(2) there exists a recurrent subset S' of states in the Markov chain M such that (2a) $|S'| \le |U_0|$ and (2b) the smallest path connecting the source state s to the recurrent subset S' has size no larger than $|U| - |U_1| - |S'|$.

If these properties hold, then by fixing $\pi(i) = u^*$ for some $i \in S'$, choosing $\pi(j), j \in S' \setminus \{i\}$ to be an arbitrary assignment into objects in U_0, and for each j in the path connecting s to S', choosing $\pi(j)$ to be an arbitrary assignment into objects in $U \setminus U_1$, we are guaranteed the following: with positive probability we reach S', and upon reaching S', the expected number of visits to u^* is infinite. Since $\nu_{u^*} > 0$, the expected utility is infinite.

To prove that these conditions are necessary for the expected utility to be infinite, observe that the only way of obtaining an infinite utility is to visit some state s' infinitely often. For this to happen, $s' \neq t$ has to be part of some recurrent subset in the Markov chain M and further, all the states in that subset have to be filled with objects with stopping probability zero (otherwise, in finite time the walk will terminate after having seen some object with non-zero stopping probability). Thus, (2a) is necessary (indeed, if all recurrent subsets not containing t have more states than the number of objects with zero stopping probability, then the random walk will have finite expected length). Furthermore, (2b) is also necessary. Indeed, if no recurrent subset S' can be filled with non-stopping objects and can be reached with positive probability, the expected length of the walk will be finite. Finally, we observe that (1) is also necessary: if each object u with $p_u = 0$ also has $\nu_u = 0$, then, while the expected length of the walk might be infinite, the utility will be zero once the walk enters the recurrent subset. Thus, the total expected utility is finite. $\square$

D.2 Proof of Theorem 3

Proof. Let $p = p_u$ for any object $u \in U$. We will reduce our problem to one where each object has stopping probability zero. To do so, we create a Markov chain M' out of the original M such that $M'_{s',s''} = (1-p)M_{s',s''}$ for each s', s'' with $s'' \neq t$. The entries of $M'_{s',t}$ will be filled so as to have a probability distribution on each row, i.e., for each s', we set $M'_{s',t} = 1 - \sum_{s'' \neq t} M'_{s',s''}$. For each s', s'' ($s'' \neq t$), the probability of transitioning from s' to s'' in the original Markov chain, with objects having stopping probability p, is equal to the probability of transitioning from s' to s'' in the new Markov chain, with objects having stopping probability zero. Thus, if we take each original object and reduce its stopping probability to zero, we have that each solution to the gap-free layout problem on the new Markov chain with the new objects has the same value it would have in the original problem, with the original objects, and the Markov chain.

To optimally solve the new problem, observe that the utility of a solution is equal to the sum, over all states s', of the expected number of visits to s' times the utility of the object in s' (i.e., $\nu_{\pi(s')}$). We consider three cases. If there exists a recurrent state different from t in M', then p was 0. Thus, the value of the solution is infinite if and only if an object with positive value exists (indeed, placing such an object in a recurrent state guarantees that our utility is lower bounded by its value times the expected number of visits to that state, which is infinite). Whether some recurrent state different from t exists in M' can be easily decided in polynomial time. If all the objects have zero utility, then each solution has zero utility. Otherwise we can assume that no recurrent state exists. In this case, it is sufficient to compute the expected number of times each state is visited in M' (which can be done by computing the inverse of $(I - M')$).

Then, the optimal solution can be found using the SORT-AND-MATCH algorithm: sort objects in decreasing order of utility, and sort states in decreasing order of expected number of visits, and match them accordingly. □

D.3 Proof of Theorem 4

Proof. By the characterization in Theorem 2, an instance has finite expected utility if and only it is possible (a) to fill a recurrent subset[4] with a set X of objects with stopping probability zero, in such a way that at least one object in X has positive utility, and (b) it is possible to fill a positive probability path from the source node s to some state in the recurrent subset with objects having stopping probability smaller than 1. In this case, then an optimal solution can be found. Otherwise, each recurrent subset is such that either (i) it contains some object with positive stopping probability, or (ii) it does not contain objects with positive utility, or (iii) the probability of getting to the recurrent subset is zero.

If the user makes more than $2^{\Theta(b)}$ steps in a recurrent subset, then each node of a recurrent subset is visited, in expectation, for at least $2^{-\Theta(b)}$ steps. Indeed, since the state s' is recurrent, it must be reachable by any node in the recurrent subset with a path $s_1, s_2, \ldots, s'$ of length upper bounded by the size of the recurrent subset, and such that each transition has probability at least $2^{-O(b_i)}$, where b_i is the number of bits used to describe the transition probabilities of node s_i. Then, the probability of getting to s' from the generic node s_1 is at most $\prod_i 2^{-O(b_i)} \geq 2^{-O(b)}$. For the same reason, the expected number of steps that a user makes outside any recurrent subset is upper bounded by $2^{O(b)}$. Therefore, if there are no recurrent subset reachable from the origin (case (iii)) the expected number of steps before termination is $2^{O(b)}$. Thus placing the maximum utility object in the initial state s, gives a $2^{O(b)}$ approximation. If we get to a recurrent subset that contains an object with positive stopping probability (case (i)), then the expected number of steps we will make in the subset is $2^{O(b)}$, since, after those many steps, we will have passed through the object with positive stopping probability (i.e., with stopping probability greater than $2^{-O(b)}$) for at least $2^{O(b)}$ times, thus ensuring that the stopping event will happen after an expected number of $2^{O(b)}$ many steps. On the other hand, if we get to a recurrent subset where each state is filled with objects of zero utility, our expected utility is 0.

In any case, the expected number of visits to states containing objects having positive utility is $2^{O(b)}$. By placing the object of maximum utility in the initial state s, we have the desired approximation guarantee. □

D.4 Proof of Proposition 5

Proof. Let $\nu^* = \max_{u \in U} \nu_u$, the maximum utility of an object, and let $u^* \in U$ be the object with this utility. Since the maximum length of a walk on the Markov chain is $d \leq n$, the maximum achievable utility is at most $d\nu^*$. By setting $\pi(s) = u^*$ and choosing the rest of the assignments arbitrarily, the expected utility is at least ν^*. □

E Proofs for Section 3

E.1 Proof of Lemma 6

Proof. Given that $\forall u \in U, p_u = 0$, the expected number of visits to each state is independent of the object assignment. Thus, to obtain an optimal solution, it suffices to sort objects in decreasing order of utility, and sort states in decreasing order of expected number of visits, and match them accordingly (we call this algorithm SORT-AND-MATCH). □

[4]Recall that $\{t\}$, while being a recurrent subset, does not satisfy this requirement since it state cannot contain any object.

E.2 Proof of Lemma 13

Proof. First,

$$\begin{aligned}
g_\pi(\Pi) &\le \sum_{i=1}^{S_\pi(\Pi)} \left(2\rho \prod_{j=1}^{i-1} \left(1 - \frac{7}{16} p_{\min}\right)^{k_{\Pi^+(j)}} \sum_{j=0}^{k_{\Pi^+(i)}-1} \left(1 - \frac{7}{16} p_{\min}\right)^j \right) \\
&\le 2\rho \sum_{i=0}^{K-1} \left(1 - \frac{7}{16} p_{\min}\right)^i = 2\rho \frac{1 - \left(1 - \frac{7}{16} p_{\min}\right)^K}{\frac{7}{16} p_{\min}} \\
&\le \frac{32}{7} \rho \frac{\min\left(\frac{7}{16} p_{\min} K, 1\right)}{p_{\min}} \le \frac{32}{7} \rho \min\left(K, \left\lceil \frac{1}{p_{\min}} \right\rceil\right),
\end{aligned}$$

where the penultimate inequality follows from $1 - (1 - 7/16 p_{\min})^K \le 1$ and $(1-a)^b \ge 1 - ab$, for $0 < a < 1 \le b$. Next, observing that $1 - (2\ln 2) p_{\min} \ge 0$, since $p_{\min} \le 1/2$ and $\ln 2 < 1$,

$$\begin{aligned}
g_\pi(\Pi) &\ge \sum_{i=1}^{S_\pi(\Pi)} \left(\frac{1}{2} \rho \prod_{j=1}^{i-1} (1 - (2\ln 2) p_{\min})^{k_{\Pi^+(j)}} \sum_{j=0}^{k_{\Pi^+(i)}-1} (1 - (2\ln 2) p_{\min})^j \right) \\
&\ge \frac{1}{2} \rho \sum_{i=0}^{K-1} (1 - (2\ln 2) p_{\min})^i \\
&\ge \frac{1}{2} \rho \sum_{i=0}^{\min\left(K, \left\lceil \frac{1}{p_{\min}} \right\rceil\right) - 1} (1 - (2\ln 2) p_{\min})^i \\
&\ge \frac{1}{2} \rho \sum_{i=0}^{\min\left(K, \left\lceil \frac{1}{p_{\min}} \right\rceil\right) - 1} (1 - (2\ln 2) p_{\min})^{\left\lceil \frac{1}{p_{\min}} \right\rceil - 1} \\
&= \frac{1}{2} \rho \min\left(K, \left\lceil \frac{1}{p_{\min}} \right\rceil\right) (1 - 2\ln 2 p_{\min})^{\frac{1}{p_{\min}}} \\
&= \frac{1}{2} \rho \min\left(K, \left\lceil \frac{1}{p_{\min}} \right\rceil\right) \left((1 - 2\ln 2 p_{\min})^{\frac{1}{2\ln 2 p_{\min}}} \right)^{2\ln 2} \\
&\ge \frac{1}{2} \rho \min\left(K, \left\lceil \frac{1}{p_{\min}} \right\rceil\right) \left((1 - \ln 2)^{\frac{1}{\ln 2}} \right)^{2\ln 2} \\
&= \frac{(1 - \ln 2)^2}{2} \rho \min\left(K, \left\lceil \frac{1}{p_{\min}} \right\rceil\right),
\end{aligned}$$

where the last inequality follows from $(1-x)^{x^{-1}}$ being a decreasing function in $x \in (0, 1]$, and $x = (2\ln 2) p_{\min} \le \ln 2$. □

E.3 Proof of Corollary 18

Proof. By the stated property of the instance, in the bucketing of Lemma 12, there will be at most t non-empty buckets. A trivial modification of its proof will then show that there will exist one bucket whose objects alone can guarantee an approximation of $O(t)$. Since all other reductions degrade the approximation factor by at most a constant, the proof follows. □

F Proofs for Section 4

F.1 Proof of Theorem 20

Proof. For simplicity of exposition, we first prove a weaker approximation bound of $O(\log^{5/2} n)$. As in the proof of Theorem 17, we bucket objects according to their utilities and stopping probabilities. Bucket $B_{i,j}$, $0 \le i, j \le O(\log n)$, contains objects u with $p_u \in (2^i, 2^{i+1}]$ and $\nu_u \in (2^{-j-1}, 2^{-j}]$. Using the same argument as in the proof of Theorem 17, we can ignore objects with utility smaller than n times the maximum object utility, and we can increase the smallest stopping probability to $1/n$, in such a way that we end up with $O(\log^2 n)$ buckets.

Consider any optimal solution and let its utility be V^*. Choose the quadrant on the grid whose states have the maximum total expected utility. Let B_{i^*,j^*} be the bucket whose objects have maximum total expected utility in the chosen quadrant. Remove the objects not in B_{i^*,j^*} from the chosen quadrant and remove all the objects from other quadrants, obtaining an assignment π. Then, the total expected utility of π will be at least $\Omega(V^*/\log^2 n)$.

Let S_i, $i \ge 0$, be the set of states at distance i from the origin in the chosen quadrant. Then, $|S_i| = i + 1$ for $0 \le i \le n/2$. Let k_i be the number of objects that are placed in S_i by π. Let $k = \sum_i k_i$.

We claim that the probability of getting to any fixed node in S_i is at most $O(1/\sqrt{i})$. To see this, sort the nodes in S_i starting from one end of the quadrant, going to the other. Suppose $i \le n/2$. For getting to the jth node in the ordering of line i in the chosen quadrant, it is necessary to get the first horizontal (left or right) and the first "vertical" (up or down) choices correct. Let us condition on this event (this does not decrease the probability of getting to j). Then, we have to make, say, $i - j$ horizontal choices and j vertical choices. Since each choice is horizontal with probability $1/2$, the probability of getting to j is the probability of getting exactly j heads in i tosses of a fair coin, i.e., $\binom{i}{j}2^{-i} < O(1/\sqrt{i})$. Thus the claim is proved for $i \le n/2$. Observe that if $i > n/2$, then the probability of getting to the jth node in the ordering is easily seen to be less than $O(1/\sqrt{i})$ (observe that the sink t becomes reachable for such a large i).

Now, fix any set S_i of states. Since the maximum probability of getting to any fixed node in S_i is at most $O(1/\sqrt{i})$, the probability of actually hitting some object in S_i is upper bounded by $\min(1, O(k_i/\sqrt{i}))$. We will show the following upper bound on the expected number of objects that will be hit:

$$O\left(\sum_{i=1}^{n}\left(\frac{\min(\sqrt{i}, k_i)}{\sqrt{i}}\right)\right) \le O\left(\sqrt{k}\right).$$

Indeed, the ith term of the sum is maximized when $\min\left(\sqrt{i}, k_i\right)$ is maximized, i.e., when $k_i = \sqrt{i}$. Furthermore, the smaller the i, the larger is the factor $i^{-1/2}$ for which the $\min(\cdot)$ term is multiplied. We can place at most k objects in the grid. Thus, choosing $k_i = \sqrt{i}$ for $i = 0, 1, \ldots, \Theta(\sqrt{k})$ (since filling $\Theta(\sqrt{k})$ levels requires $\Theta(k)$ objects) maximizes the expected number of objects that will be hit, and gives the claimed upper bound.

We now give a different upper bound on the expected number of objects that will be hit. We will use the smallest of the two in the following. Since we are considering objects in B_{i^*,j^*}, their stopping probability is at least 2^{-j^*-1}. Thus, in expectation, we will hit at most $O\left(2^{j^*}\right)$ of them. Then, the maximum expected utility of the quadrant, given π, is $O\left(2^{i^*}\min\left(2^{j^*}, \sqrt{k}\right)\right)$. By the choice of B_{i^*,j^*} the expected utility of the optimal assignment can be upper bounded by $O((\log^2 n)2^{i^*}\min\left(2^{j^*}, \sqrt{k}\right)$.

We now show how to fill the quadrant with objects from B_{i^*,j^*} in such a way that the expected utility is

$$\Omega\left(2^{i^*}\min\left(2^{j^*}, \sqrt{k/\log k}\right)\right).$$

Furthermore, if the empty states in the chosen quadrant and in the other quadrants are filled arbitrarily with the remaining objects, then the expected utility is reduced only by a constant. This leads to an $O(\log^{5/2} n)$ approximate solution to the gap-free Markov layout problem on grids.

First, guess the best bucket (since there are $O(\log^2 n)$ buckets, we can just enumerate all of them). For each $i \ge 0$, as long as there are available objects, place $\sqrt{i \log k}$ objects in the states of S_i, uniformly around the origin (i.e, fill with objects from B_{i^*,j^*} the $\sqrt{i \log k}/2$ states just at the left of the origin, and do the same for the $\sqrt{i \log k}/2$ states just at the right of the origin.) The process will go on for $\Theta(\sqrt{k/\log k})$ levels S_i, $i = 1, \ldots, \Theta(\sqrt{k/\log k})$, since $\Theta(\sqrt{k \log k} \cdot \sqrt{k/\log k}) = \Theta(k)$.

By the Chernoff bound, the probability of hitting some state in the first $\Theta(\sqrt{k/\log k})$ levels that is not filled with objects from B_{i^*,j^*} is $O(k^{-2})$, if the walk enters the right quadrant. The probability of entering the right quadrant is $1/4 - o(1)$. Thus, with probability $1/4 - O(k^{-2})$, we do not hit any state in the first $\Theta(\sqrt{k/\log k})$ levels that is not filled with objects from B_{i^*,j^*}.

If this does not happen, then the probability of stopping because of some object stopping event, before having seen $\Omega\left(\min\left(2^{j^*}, \sqrt{k/\log k}\right)\right)$ objects is less than an (arbitrarily small) constant. Thus, we will obtain a utility of $\Omega\left(2^{i^*} \min\left(2^{j^*}, \sqrt{k/\log k}\right)\right)$ with constant probability, and the expected utility will thus be $\Omega\left(2^{i^*} \min\left(2^{j^*}, \sqrt{k/\log k}\right)\right)$. Observe that we can fill the states that are still empty with any objects without decreasing the lower bound we just obtained on the utility of our solution.

We can improve the approximation bound to $O(\log^{3/2} n)$. To do so, one needs to bucket only according to the object probabilities, and then show that it is always better to put objects with higher utilities in lower levels. We omit the details of this improvement from this version. $\square$

G Proofs for Section 5

G.1 Proof of Theorem 21

Proof. We reduce from the CUBIC MAX-BISECTION problem: given a 3-regular undirected graph $G = (V, E)$, find $S \subset V$, $|S| = |V|/2$ that maximizes the number of edges in the cut. As observed by Feige et al. [9], the hardness result for CUBIC MAX-CUT of [3] implies that, if P≠NP then CUBIC MAX-BISECTION cannot be approximated to a ratio $\geq \frac{331}{332} + \epsilon = \rho$, for each $\epsilon > 0$.

We let $n = |V|$, and we create a Markov chain on the states $S = V \cup \{s, t\}$. For each edge $\{v_i, v_j\}$ in G, we create two edges (v_i, v_j) and (v_j, v_i) in the Markov Chain. The source node s will have an edge to each node $v_i \in V$. For each node $v_i \in V$, we will add an edge (v_i, t). The transition from s to v_i, for each $v_i \in V$, will happen with probability $1/n$. The transition from a node $v_i \in V$ to t will happen with probability $1 - \alpha$, to be fixed later. If $\{v_i, v_j\} \in E$ the transition from v_i to v_j will happen with probability $\frac{\alpha}{3}$.

The instance will contain $n/2$ (n is even, since G is cubic) *good* objects, each good object u having $p_u = 1 = \nu_u$ and a *starting* object u_s with $p_{u_s} = 0, \nu_{u_s} = 1$. If we insist on the instance to have at least as many objects as fillable states, we also add $n/2$ *dummy* objects u' each with $p_{u'} = 0 = \nu_{u'}$. We observe that if an assignment places a dummy object in a state, the objective value remains the same if the dummy object is removed.

Let $S^* \subset V$ be the solution to the CUBIC MAX-BISECTION problem and let t be the value of the solution. By filling the nodes in S^* with good objects and by placing the starting object in s, one can obtain a total utility of at least

$$1 + \frac{n/2}{n} + \frac{n/2}{n} \alpha \frac{2t}{3n}.$$

Indeed, the object in s will result in a utility of 1. The transition out of s will be uniform over V. Therefore with probability $1/2$, the user will land on some state of S^*, finding a good object, accruing a utility of 1, and stopping the random walk; otherwise, again with probability $1/2$, the user will land on a uniform at random empty state in $V \setminus S^*$. Since the graph is 3-regular and since t edges go out of $V \setminus S^*$, the probability that the user will make a move to S^* (thus accruing a utility of 1, thanks to the node there) will be $\alpha t \left(3\frac{n}{2}\right)^{-1}$. Since we only care about a lower bound on the total utility, and since no node has negative utility, we do not consider what happens if the user happens to move to another state in $V \setminus S^*$.

Suppose instead that the maximum number of edges between a set S^* of $n/2$ nodes and $V \setminus S^*$ is at most ρt. We start by proving that if the starting object is not in s, then we can place it in s without decreasing the total utility. Indeed,

- if s contained a good object, then the total utility was exactly 1, and by placing the starting object in s, the total utility is at least 1;

- if s was empty or contained a dummy object, then placing the starting object in s will increase the utility accrued thanks to the starting object, and will not decrease the utility accrued with any other object. Observe that the probability of visiting k nodes of V (possibly with repetitions) is upper bounded by α^{k-1}. Furthermore, the probability of visiting any single state of V as the first after s is $\frac{1}{n}$. It follows that, if the starting object was in some state in V, then the expected number of visits to the starting object is at most

  $$\frac{1}{n}+\sum_{i=1}^{\infty}\alpha^i=\frac{1}{n}+\frac{\alpha}{1-\alpha},$$

  which is less than 1 if $\alpha<\frac{1}{3}$ and $|V|\geq 4$ (which has to be true in any non-empty cubic graph). On the other hand, if the starting object in placed in s, then the number of visits to that object is deterministically 1. Observing that emptying the state of V that contained s does not decrease the probability of any walk concludes the proof of this step.

Now, we show that the total utility of an assignment that does not use all the $n/2$ good objects on the states of V is not decreased by placing arbitrarily the unused good objects in states of V that are either empty or that contain dummy objects. Indeed, after we accrue the unit utility thanks to the starting object in s, two things can happen: either we hit a good object in some node of V, accruing another utility of 1 and stopping the walk, or we do not, accruing no new utility and finally ending in the sink state. Therefore adding a good objects to a node $v\in V$ cannot decrease the total utility: every walk that passed through v before hitting a state with a good object, will now give a utility of 1 thanks to the good object in v and in no other case can the utility decrease. We can therefore assume that all the good objects are used in the assignment.

We finally show that if the starting object is in s, and each good object is assigned, we have that the total utility is at most

$$1+\frac{n/2}{n}+\frac{n/2}{n}\alpha\frac{2\rho t}{3n}+\frac{n/2}{n}\alpha^2.$$

Indeed, we accrue a unit utility thanks to the starting object; then we will move to a state containing a good object, accruing another utility of 1 and stopping, with probability $\frac{n/2}{n}=\frac{1}{2}$. Otherwise, with probability $\frac{1}{2}$ we will move to a uniformly at random empty state (or to a state filled with a dummy object); from there, to accrue a unit utility, we can either (i) make a step towards a state containing a good object (with probability at most $\alpha\frac{2t}{3n}$), or (ii) move to another empty (or filled with a dummy object) state, and after a non-negative number of other steps, finish our walk a state with a good object. The, the upper bound follows by observing that in case (ii), we have to avoid a transition to t at least two times.

We now compare the two bounds. If a max-bisection of value t exists, then the total utility is at least $\frac{3}{2}+\frac{\alpha t}{3n}$. If the max-bisection has value at most ρt, then by using the lower bound of $t\geq\frac{|E|}{2}$ on the maximum bisection of a graph $G(V,E)$ (see [12]), which simplifies to $t\geq\frac{3n}{4}$ in a cubic graph, we get that the total utility is at most

$$\begin{aligned}\frac{3}{2}+\frac{\alpha\rho t}{3n}+\frac{\alpha^2}{2}&=\frac{3}{2}+\frac{\alpha t}{3n}+\left(\frac{\alpha^2}{2}-\frac{(1-\rho)\alpha t}{3n}\right)\leq\frac{3}{2}+\frac{\alpha t}{3n}+\left(\frac{\alpha^2}{2}-\frac{(1-\rho)\alpha}{4}\right)\\&=\frac{3}{2}+\frac{\alpha t}{3n}+\frac{\alpha}{4}\cdot(2\alpha+\rho-1)\leq\frac{3}{2}+\frac{\alpha t}{3n}-\frac{\alpha}{4000},\end{aligned}$$

where the last step follows from $\rho\leq 1-\frac{3}{1000}$, and the choice $\alpha=\frac{1}{1000}$.

Therefore, since in a cubic graph we have $|E|\leq\frac{3}{2}n$, and since $t\leq|E|$, the inapproximability ratio can be expressed as

$$\frac{\frac{3}{2}+\frac{\alpha t}{3n}}{\frac{3}{2}+\frac{\alpha t}{3n}-\frac{\alpha}{4000}}\geq\frac{\frac{3}{2}+\frac{\alpha}{2}}{\frac{3}{2}+\frac{\alpha}{2}-\frac{\alpha}{4000}}=\frac{6002000}{6001999}=1+\frac{1}{6001999}.$$

□

G.2 Proof of Theorem 22

Proof. We reduce from the BIPARTITE-CLIQUE problem: given an undirected bipartite graph $G = (V, W, E)$, and a positive integer k, does there exist $V' \subseteq V, W' \subseteq W$, $|V'| = |W'| = k$, such that V' and W' induce a bipartite clique in G? We assume that each node in V has the same degree Δ, as this assumption leaves the problem NP-hard (indeed, if Δ is the maximum degree in V, we could just add, for each $u \in V$, $\Delta - \deg(u)$ new nodes to W and connect each of them to v).

We modify G by directing edges in from V to W, obtaining a directed graph H. We use the term *source* to denote the nodes in V and the term *sink* to denote the nodes in W.

For a τ to be fixed later, we then create $\ell = 2\tau(k^2 - 1)$ copies $H_1, \ldots, H_\ell$ of H, and we add $\ell + 1$ new nodes $x_0, x_1, \ldots, x_\ell$. For each $0 \le i < \ell$, we add an edge from x_i to each source node in H_{i+1} and for each $0 < i \le \ell$, we add an edge from each sink node in H_i to x_i. Finally the node x_ℓ will have a self-loop. This defines the underlying graph of our Markov chain. Let $n = \ell(|V| + |W| + 1) + 1$ be the number of its nodes. (Observe that the graph is a DAG, except for the self-loop on x_ℓ.)

The transition probabilities of the Markov chain are as follows: each node except for x_ℓ will choose one of its out-neighbors uniformly at random. From x_ℓ, there is a transition to itself (self-loop) with probability $p = 1 - (|V|\,\Delta)^{-\ell}$, and with the remaining problem, the walk ends.

To finish the description of the Markov layout problem instance, we describe the set of objects. Each object will have expected utility of 1. There will be $g = \ell(2k + 1) + 1$ *good* objects, having stopping probability 0, and $b = n - \ell(2k + 1) + 1$ *bad* objects with stopping probability 1. Observe that since we require the solution to be gap-free, a valid solution needs to use each single object, i.e., $n = b + g$.

We claim that it is NP-hard to distinguish between instances of utility at least $k^{2\ell}$ and instances of utility at most $(k^4 - k^2)^{\ell/2} + n$. Assuming this (Lemma 26), we finish the proof.

For $\tau \ge \ln n$, the latter is at most $\le 2(k^4 - k^2)^{\ell/2}$. Thus, the inapproximability bound becomes

$$\begin{aligned}\frac{k^{2\ell}}{2(k^4 - k^2)^{\ell/2}} &\ge \frac{1}{2}\left(\frac{k^4}{k^4 - k^2}\right)^{\ell/2} = \frac{1}{2}\left(1 + \frac{1}{k^2 - 1}\right)^{\ell/2} \\ &= \frac{1}{2}\left(\left(1 + \frac{1}{k^2 - 1}\right)^{k^2-1}\right)^{\tau} \ge \frac{1}{2}2^{\tau}.\end{aligned}$$

Here, the last inequality is justified since $k \ge 2 \implies \left(1 + \frac{1}{k^2-1}\right)^{k^2-1} \ge 2$.

Observe that we can choose any τ polynomial in $|V| + |W|$. Choosing $\tau = (|V| + |W|)^{\frac{1}{\epsilon}}$, gives us $\ell \le O((|V| + |W|)^{\frac{1}{\epsilon}+2})$ and $n \le O((|V| + |W|)^{\frac{1}{\epsilon}+3}) = O((\tau)^{1+3\epsilon})$. Then, $\tau \ge n^{1-O(\epsilon)}$ and the inapproximability bound becomes $2^{n^{1-O(\epsilon)}}$, which completes the proof. ☐

Lemma 26. *It is NP-hard to distinguish whether an instance has utility at least* $k^{2\ell}$ *or utility at most* $(k^4 - k^2)^{\ell/2} + n$.

Proof. Suppose that the original graph contained a $k \times k$ bipartite clique. Let $V' \subseteq V, W' \subseteq W$ be the two maximal independent sets of the bipartite clique.

Consider the following assignment. A good object is assigned to each node x_i, $i = 0, \ldots, \ell$, and to each of the ℓ copies of V' and W'. Bad objects are assigned to the other nodes. Then the probability of getting from some x_i to x_{i+1} will then be $\frac{k}{|V|}\frac{k}{\Delta}$ (the probability of ending up in V' from x_i times the probability of getting from there to W'). Then, the probability of getting to x_ℓ at least once will be $k^{2\ell}\left(|V|\,\Delta\right)^{-\ell}$.

Once we arrive at x_ℓ for the first time, we compute the expected number of times will we cycle through its loop: it is $\sum_{i=1}^{\infty} p^i = \frac{p}{1-p}$. Since each time we transition to x_ℓ we accrue a unit utility, the total expected utility from node x_ℓ, conditioned on getting there, is $1 + \frac{p}{1-p} = \frac{1}{1-p} = \left(|V|\,\Delta\right)^{\ell}$. The expected total utility of x_ℓ is then $k^{2\ell}$. Since the expected total utility of any single node is a lower bound on the expected total utility of the Markov chain, we have showed that if there exists a solution to the bipartite clique problem, then the best solution for the Markov layout problem has utility at least $k^{2\ell}$.

Now suppose that the original graph did not contain a $k \times k$ bipartite clique. We show that the maximum expected utility of the Markov layout problem is at most $(k^4 - k^2)^{\ell/2} + n$. Consider any assignment and observe that for it to have utility more than $(k^4 - k^2)^{\ell/2} + n$, it is necessary that each node x_i is assigned a good object — in fact, if this does not happen then the expected utility of node x_ℓ is zero. Since node x_ℓ is the only node that can be visited more than once, if some x_i is not assigned a good object, then the maximum expected utility is upper-bounded by n, the number of states times the maximum utility of an object. Therefore, we assume that each x_i, $i = 0, \ldots, \ell$, is assigned a good object.

Observe that a path from x_0 to x_ℓ has either probability 0, or probability $(|V|\,\Delta)^{-\ell}$, of being followed. In particular, the latter case occurs if and only if it is only composed of good objects. Therefore, the expected utility of object x_ℓ is linear in the number of such paths, which we call *useful* paths. Thus, given an assignment, the expected utility of node x_ℓ is linear in the number of useful paths.

Fix any copy H_s in the instance, and consider the number e_s of its edges that are hit on both sides by a good object. Then, the number of useful paths is $\prod_{s=1}^{\ell} e_s$. Observe that, if a generic copy H_s has i good objects on one side and j good objects on the other, then $e_s \le ij$. Further, observe that if $i = j = k$, then the stronger bound of $e_s \le k^2 - 1$ holds, since we assumed the original graph not to contain a bipartite $k \times k$ clique.

Let $C_{i,j}$, $i, j \ge 0$, be the number of copies of H in the instance that are assigned i good objects on the "V-side" and j good objects on the "W-side". Then, $\sum_{i=0}^{|V|} \sum_{j=0}^{|W|} C_{i,j} = \ell$. The number of useful paths is upper bounded by

$$u = \left(k^2 - 1\right)^{H_{k,k}} \prod_{\substack{i,j \ge 1 \\ i \ne k \,\vee\, j \ne k}} (ij)^{H_{i,j}}\,.$$

Consider only the bipartite copies that are not assigned k objects both on their V and their W sides. The overall number of such bipartite copies will be $\ell - H_{k,k}$. Let H_i be the number of sides, of those bipartite copies, that are assigned exactly i objects. Then, $\sum_{i \ge 0} H_i = 2(\ell - H_{k,k})$, since each copy, but the $H_{k,k}$ ones, contributes exactly two sides. We have that $H_k \le \ell - H_{k,k}$ since each of those bipartite copies has at most one side with k good objects. Observe that

$$\prod_{\substack{i,j \ge 1 \\ i \ne k \,\vee\, j \ne k}} (ij)^{H_{i,j}} = \prod_{i \ge 1} i^{H_i}.$$

Further, observe that $(k^2 - 1)^{H_{k,k}} = (k+1)^{H_{k,k}}(k-1)^{H_{k,k}}$. Then, if we define

$$H'_i = \begin{cases} H_i + H_{k,k} & \text{if } i \in \{k-1, k+1\}, \\ H_i & \text{otherwise,} \end{cases}$$

we obtain that $\sum_{i \ge 1} H'_i = 2\ell$, and

$$u = \prod_{i \ge 1} i^{H'_i}. \tag{1}$$

Observe that $H'_k = H_k \le \ell - H_{k,k} \le \ell$, so that $\sum_{i \ne k} H'_i \ge \ell$. Our objective is to show that $u \le k^{2\ell}\left(1 - \frac{1}{k^2}\right)^{\ell/2}$. To this end, we will substitute pairs of terms $\langle a, b\rangle$ (such that $a \le k-1$ and $b \ge k+1$) in the product of (1) with terms $\langle a+1, b-1\rangle$, showing that this will decrease the value of the product by a factor of $1 - \frac{1}{k^2}$. We will repeat this substitution as long as such as pair $\langle a, b\rangle$ exists. Since $\sum_{i \ne k} H'_i \ge \ell$, this will happen at least $\ell/2$ times, showing that $u \le k^{2\ell}\left(1 - \frac{1}{k^2}\right)^{\ell/2}$.

So it remains to be shown that, if $a \le k-1$ and $b \ge k+1$, then $ab \le \left(1 - \frac{1}{k^2}\right)(a+1)(b-1)$. We show it with the following chain of inequalities:

$$\begin{aligned} \frac{ab}{(a+1)(b-1)} &= 1 - \frac{b-a-1}{(a+1)(b-1)} \le 1 - \frac{b-k}{k(b-1)} \\ &= 1 - \frac{1 - \frac{k}{b}}{k - \frac{k}{b}} \le 1 - \frac{1 - \frac{k}{k+1}}{k - \frac{k}{k+1}} = 1 - \frac{1}{k^2}. \end{aligned}$$

The expected utility of node x_ℓ will then be $u\left(|V|\,\Delta\right)^{-\ell}\frac{1}{1-p} = u \leq k^{2\ell}(1-k^{-2})^{\ell/2}$. Since each other node can produce a utility of at most 1 (since it will be visited at most once), we have that if the original graph did not contain a $k \times k$ bipartite clique, then the maximum expected utility is at most $(k^4 - k^2)^{\ell/2} + n$. The proof is complete. □

H Proofs for Section A

H.1 Proof of Theorem 23

Proof. First, we bucket the objects according to their stopping probabilities as in Theorem 17. Consider any optimal assignment and remove all the objects from buckets that are suboptimal (in terms of total expected utility of objects in the bucket). Suppose the probability bucket $(p/2, p]$ survives this step. In the tree, remove all the nodes of height more than $1/p$ and evaluate the probabilities of each remaining state. Apply the SORT-AND-MATCH algorithm to the objects in the surviving bucket and arbitrarily assign other objects to the remaining states (i.e., even the ones that have been removed). We claim that the obtained solution is an $O(\log n)$ approximation to the gap-free Markov layout problem on trees.

To see this, first note that in expectation, no more than $O(1/p)$ objects of the surviving bucket could be seen. Also note that a parent state has no lower probability of being seen than any of its children. Therefore, we can conclude that the surviving bucket's objects form a connected component (containing the root) in the tree. Thus, adding other objects does not decrease the probability of seeing any of the chosen bucket's objects, and thus of the expected utility of the solution. Finally, we observe that the probability of stopping at least once in $1/p$ object stopping events is a constant $c < 1$. This lets us conclude that the expected utility of assignment of the chosen bucket is a constant approximation to their optimum. □

H.2 Proof of Theorem 24

Proof. We bucket objects according to both their stopping probabilities and their utilities, as in Theorem 17. Each of the $B = O(\log^2 n)$ buckets will contain at most a linear number of objects. Furthermore, as in Theorem 17, we remove all objects with utility less than the maximum utility times ϵ/n, and we round both the stopping probability and the utility of each object to the nearest power of $1+\epsilon$. As we already argued, this does not change the value of the optimal solution by more than $1 \pm O(\epsilon)$.

We then construct a dynamic program, computing quantities bottom-up. For each node v of the tree, we maintain a table T_v of B dimensions, where each dimension corresponds to a bucket. The entry $T_v(k_1, \ldots, k_B)$ of v's table will contain the maximal expected utility of the subtree rooted at v, if one were to use k_1 objects from the first bucket, $\cdots$, k_B objects from the Bth bucket.

If v is a leaf, then $T_v(e_i)$ contains the utility of bucket i (if it is non-empty) and $-\infty$ otherwise (denoting infeasible); all other entries of T_v are set to $-\infty$. Here, e_i is the binary vector with a 1 in the ith coordinate,

If v is an internal node, we fill its table in the following way. We first initialize $T_v(\cdot) \leftarrow -\infty$ and $T_v(0, \ldots, 0) \leftarrow 0$.

For each child w of v, for each $(k_1, \ldots, k_B)$, and for each $(\ell_1, \ldots, \ell_B)$, we set

$$\begin{aligned}
&T_v(k_1+\ell_1, \ldots, k_B+\ell_B) \leftarrow \\
&\qquad \max\{T_v(k_1+\ell_1, \ldots, k_B+\ell_B), \\
&\qquad\quad T_v(\ell_1, \ldots, \ell_B) + T_w(k_1, \ldots, k_B)\}.
\end{aligned}$$

Then for each bucket i, (we assume that the object placed in v came from bucket i), for each position $(\ell_1, \ldots, \ell_i, \ldots \ell_B)$, we set

$$\begin{aligned}
&T_v(\ell_1, \ldots, \ell_{i-1}, \ell_i+1, \ell_{i+1}, \ldots, \ell_B) \leftarrow \\
&\qquad \max\{T_v(\ell_1, \ldots, \ell_{i-1}, \ell_i+1, \ell_{i+1}, \ldots, \ell_B), \\
&\qquad\quad T_v(\ell_1, \ldots, \ell_{i-1}, \ell_i, \ell_{i+1}, \ldots, \ell_B)(1-p_i) + \nu_i\},
\end{aligned}$$

where p_i is the stopping probability of (the objects in) bucket i and ν_i is the utility of bucket i.

Then, by looking at the table of the root of the tree, we can return a $(1+\epsilon)$-approximation to the best solution value. Specifically, we look at those cells whose coordinates (i) are feasible given the buckets sizes (for a bucket X_i, we can use at most $|X_i|$ many objects from the bucket), and (ii) whose sum of coordinates equals the number of nodes in the tree. Note that (ii) is needed only for the gap-free version. We defer the details to the full version of this paper. $\square$