On the Approximability of Dynamic Min-Sum Set Cover

Dimitris Fotakis
National Technical University of Athens

Panagiotis Kostopanagiotis
National Technical University of Athens

Vasileios Nakos
Saarland University and Max Planck Institute for Informatics

Georgios Piliouras
Singapore University of Technology and Design

Stratis Skoulakis
Singapore University of Technology and Design

Abstract

We investigate the polynomial-time approximability of the dynamic version of Min-Sum Set Cover (Dyn-MSSC), a natural and intriguing generalization of the classical List Update problem. In Dyn-MSSC, we maintain a sequence of permutations $(\pi^0, \pi^1, \ldots, \pi^T)$ on n elements, based on a sequence of requests $\mathcal{R} = (R^1, \ldots, R^T)$. We aim to minimize the total cost of updating π^{t-1} to π^t, quantified by the Kendall tau distance $\mathrm{d}_{\mathrm{KT}}(\pi^{t-1}, \pi^t)$, plus the total cost of covering each request R^t with the current permutation π^t, quantified by the position of the first element of R^t in π^t.

Using a reduction from Set Cover, we show that Dyn-MSSC does not admit an $O(1)$-approximation, unless $\mathrm{P} = \mathrm{NP}$, and that any $o(\log n)$ (resp. $o(r)$) approximation to Dyn-MSSC implies a sublogarithmic (resp. $o(r)$) approximation to Set Cover (resp. where each element appears at most r times). Our main technical contribution is to show that Dyn-MSSC can be approximated in polynomial-time within a factor of $O(\log^2 n)$ in general instances, by randomized rounding, and within a factor of $O(r^2)$, if all requests have cardinality at most r, by deterministic rounding.

2012 ACM Subject Classification Theory of Computation → Design and Analysis of Algorithms

Keywords and phrases Approximation Algorithms, Dynamic Min-Sum Set Cover, Dynamic Optimization Problems

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding *Dimitris Fotakis*: Partially supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under project BALSAM, HFRI-FM17-1424.
Vasileios Nakos: Supported by the project TIPEA that has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No. 850979).
Georgios Piliouras: Supported by NRF2019-NRF-ANR095 ALIAS grant, grant PIE-SGP-AI-2018-01 and NRF 2018 Fellowship NRF-NRFF2018-07.
Stratis Skoulakis: Supported by NRF 2018 Fellowship NRF-NRFF2018-07.

1 Introduction

In *Dynamic Min-Sum Set Cover* (Dyn-MSSC), we are given a universe U on n elements, a sequence of requests $\mathcal{R} = (R_1, \ldots, R_T)$, with $R_t \subseteq U$, and an initial permutation π^0 of the elements of U. We aim to maintain a sequence of permutations $(\pi^0, \pi^1, \ldots, \pi^T)$ of U, so as to minimize the total cost of updating (or moving from) π^{t-1} to π^t in each time step plus the total cost of covering each request R_t with the current permutation π^t. The cost of moving from π^{t-1} to π^t is the number of inverted element pairs between π^{t-1} and π^t, i.e., the Kendall Tau distance $\mathrm{d_{KT}}(\pi^{t-1}, \pi^t)$. The cost $\pi^t(R_t)$ of covering a request R_t with a permutation π^t is the position of the first element of R_t in π^t, i.e., $\pi^t(R_t) = \min\{i \mid \pi^t(i) \in R_t\}$. Thus, given $\mathcal{R} = (R_1, \dots, R_T)$, we aim to minimize $\sum_{t=1}^{T} \big(\mathrm{d_{KT}}(\pi^{t-1}, \pi^t) + \pi^t(R_t)\big)$.

The Dyn-MSSC problem is a natural generalization of the (offline version of the) classical *List Update* problem [26], where $|R_t| = 1$ for all requests $R_t \in \mathcal{R}$. The offline version of List Update is NP-hard [2], while it is known that any 5/4-approximation has to resort to *paid exchanges*, where an element different from the requested one is moved forward to the list [24, 28]. Dyn-MSSC was introduced in [17] as the dynamic extension of Min-Sum Set Cover (MSSC) [15], where we aim to compute a single static permutation π that minimizes the total covering cost $\sum_{t=1}^{T} \pi(R_t)$. [17] presented a (simple polynomial-time) online algorithm for Dyn-MSSC with competitive ratio between $\Omega(r\sqrt{n})$ and $O(r^{3/2}\sqrt{n})$ for r-bounded instances, where all requests have cardinality at most r, and posed the polynomial-time approximability of Dyn-MSSC as an interesting open question. Dyn-MSSC is also related to recently studied time-evolving (a.k.a. multistage or dynamic) optimization problems (e.g., multistage matroid, spanning set and perfect matching maintenance [19], time-evolving Facility Location [14, 3]), where we aim to maintain a sequence of near-optimal feasible solutions to a combinatorial optimization problem, in response to time-evolving underlying costs, without changing too much the solution from one step to the next.

Motivation. Dyn-MSSC is motivated by applications, such as web search, news, online shopping, paper bidding, etc., where items are presented to the users sequentially. Then, the item ranking is of paramount importance, because user attention is usually restricted to the first few items in the sequence (see e.g., [27, 13, 16, 10]). If a user does not spot an item fitting her interests there, she either leaves the service (in case of news or online shopping, see e.g., the empirical evidence in [12]) or settles on a suboptimal action (in case of paper bidding, see e.g., [11]). To mitigate such situations and increase user retention, modern online services highly optimize item rankings based on user scrolling and click patterns. Each user t is represented by her set of preferred items (or item categories) R_t . The goal of the service provider is to continually maintain an item ranking π^t, so that the current user t finds one of her favorite items at a relatively high position in π^t. Continual ranking update is dictated by the fact that users with different characteristics and preferences tend to use the online service during the course of the day (e.g., elderly people in the morning, middle-aged people in the evening, young people at the night – similar patterns apply for people from different countries and timezones). Moreover, different user categories react in nonuniform ways to different trends (in e.g., news, fashion, sports, scientific topics). For consistency and stability, however, the ranking should change neither too much nor too frequently. Dyn-MSSC makes the (somewhat simplifying) assumptions that the service provider has a relatively accurate knowledge of user preferences and their arrival order, and that its total cost is proportional to how deep in π^t the current user t should reach, before she finds one of her favorite items, and to how much the ranking changes from one user to the next.

From a theoretical viewpoint, Dyn-MSSC was used in [17] as a natural benchmark for studying the dynamic competitive ratio of Online Min-Sum Set Cover, where the algorithm updates its permutation online, without any knowledge of future requests. As in Dyn-MSSC, the objective is to minimize the total moving plus the total covering cost.

Contribution and Techniques. In this work, we initiate a study of the polynomial-time approximability of Dyn-MSSC. Using a reduction from Set Cover, we show (Theorem 7) that Dyn-MSSC does not admit a $c \log n$-approximation, for some absolute constant c, unless P = NP. Moreover our reduction establishes that an $o(r)$-approximation for r-bounded instances of Dyn-MSSC implies an $o(r)$-approximation for Set Cover, in case each element appears in at most r requests.

Our main technical contribution is to show that Dyn-MSSC can be approximated in polynomial-time within a factor of $O(\log^2 n)$ in general instances, by randomized rounding (Theorem 10), and within a factor of $O(r^2)$ in r-bounded instances, by deterministic rounding (Theorem 11).

For both results, we consider a restricted version of Dyn-MSSC, inspired by the Move-to-Front (MTF) algorithm for List Update, where in each time step t, we can only move a single element of R_t from its position in π^{t-1} to the first position of π^t. Since such a permutation π^t coves R_t with unit cost, we now aim to select the element of each R_t moved to front of π^t, so as to minimize the total moving cost $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{KT}}(\pi^{t-1}, \pi^t)$. It is not hard to see that the optimal cost of serving $\mathcal{R}$ under the restricted Move-to-Front version of Dyn-MSSC is within a factor of 4 from the optimal cost under the original, more general, definition of Dyn-MSSC.

Hence, approximating Dyn-MSSC boils down to determining which element of R_t should become the top element of π^t. To this end, we relax permutations to doubly stochastic matrices and consider a Linear Programming relaxation of the restricted Move-to-Front version of Dyn-MSSC, which we call *Fractional-MTF* (see Definition 8). Given the optimal solution of the aforementioned linear program, which is a sequence of doubly stochastic matrices $(A^0, A^1, \ldots, A^T)$, with A^0 corresponding to the initial permutation π^0, our main technical challenge is to round each doubly stochastic matrix A^t to a permutation π^t such that (i) there is an element of R_t at one of the few top positions of π^t; and (ii) the total moving cost $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{KT}}(\pi^{t-1}, \pi^t)$ of the rounded solution is comparable to the total moving cost $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(A^{t-1}, A^t)$ of the optimal solution of Fractional-MTF, where d_{FR} is a notion of distance equivalent to Spearman's footrule distance on permutations (see Definition 4).

Working towards a randomized rounding approach, we first observe that rounding each doubly stochastic matrix independently may result in a permutation sequence with total moving cost significantly larger than that of Fractional-MTF (see also the discussion after Lemma 9). In Theorem 10, we show that a dependent randomized rounding with logarithmic scaling of entries (Algorithm 1), similar in spirit with the randomized rounding approach [8, 25] for Generalized Min-Sum Set Cover, results in an approximation ratio of $O(\log^2 n)$. Interestingly, Algorithm 1 without the logarithmic scaling results in a permutation sequence with the expected moving cost within a factor of 4 from the optimal moving cost of Fractional-MTF. However, we lose a logarithmic factor in the approximation ratio, because we need to scale up the entries of each doubly stochastic matrix A^t, so as to ensure that some element of R_t appears in the few top positions of π^t with sufficiently large probability. The other logarithmic factor is lost because there could be a logarithmic number of elements allocated to the same position of the resulting permutation by the randomized rounding.

Our deterministic rounding of Algorithm 2 for r-bounded request sequences is motivated by the deterministic rounding for Set Cover and Vertex Cover. We observe that in the optimal solution of Fractional-MTF, in each time step t, there is some element $e \in R_t$ with $A^t_{e1} \geq 1/r$ (i.e., e occupies a fraction of at least $1/r$ of the first position in the "fractional permutation" A^t). Algorithm 2 simply moves any such element to the front of π^t. The most challenging part of the analysis is to establish that for any optimal solution $(A^0, A^1, \ldots, A^T)$ of Fractional-MTF with respect to an r-bounded request sequence, there exists a sequence of doubly stochastic matrices $(A^0, \hat{A}^1, \ldots, \hat{A}^T)$ with the entries of each $\hat{A}^t$ being multiples of $1/r$, such that (i) the moving cost of $(A^0, \hat{A}^1, \ldots, \hat{A}^T)$ is bounded from above by the optimal cost of Fractional-MTF; and (ii) each matrix $\hat{A}^t$ contains in the first position the element that Algorithm 2 keeps in the first position at round t, with mass at least $1/r$. Then we show

(Lemma 20) that for any sequence of doubly stochastic matrices $(A^0, \hat{A}^1, \ldots, \hat{A}^T)$ satisfying the above properties, the moving cost of Algorithm 2 is at most the moving cost of the doubly stochastic matrices, $\sum_{t=1}^{T} \mathrm{d_{FR}}(\hat{A}^t, \hat{A}^{t-1})$. The latter is done through the use of an appropriate potential function based on an extension of the Kendall-Tau distance to doubly stochastic matrix with entries being multiples of $1/r$.

A potentially interesting insight is that the technical reason for the quadratic dependence of our approximation ratios on $\log n$ and r is conceptually similar to the reason for the (best possible) approximation ratio of $4 = 2 \cdot 2$ in [15] (see the discussion after Theorem 10). Hence, we conjecture that any $o(\log^2 n)$ (resp. $o(r^2)$) approximation to Dyn-MSSC must imply a sublogarithmic (resp. $o(r)$) approximation to Set Cover.

Other Related Work. The MSSC problem generalizes various NP-hard problems, such as Min-Sum Vertex Cover and Min-Sum Coloring and it is well-studied. Feige, Lovasz and Tetali [15] proved that the greedy algorithm, which picks in each position the element that covers the most uncovered requests, is a 4-approximation (that was also implicit in [9]) and that no $(4-\varepsilon)$-approximation is possible, unless P = NP. In Generalized MSSC (a.k.a. *Multiple Intents Re-ranking*), there is a covering requirement $K(R_t)$ for each request R_t and the cost of covering a request R_t is the position of the $K(R_t)$-th element of R_t in the (static) permutation π. The MSSC problem is the special case where $K(R_t) = 1$ for all requests R_t. Another notable special case of Generalized MSSC is the Min-Latency Set Cover problem [20], which corresponds to the other extreme case where $K(R_t) = |R_t|$ for all requests R_t. Generalized MSSC was first studied by Azar et al. [5], who presented a $O(\log r)$-approximation; later $O(1)$-approximation algorithms were obtained [8, 25, 23, 6].

Further generalizations of Generalized MSSC have been considered, such as the Submodular Ranking problem, studied in [4], which generalizes both Set Cover and MSSC, and the Min-Latency Submodular Cover, studied by Im et al. [22]. We refer to [22, 21] for a detailed discussion on the connections between these problems and their applications.

The online version of MSSC, which generalizes the famous List Update problem, was studied in [17]. They proved that its static deterministic competitive ratio is $\Theta(r)$ and presented a natural memoryless algorithm, called *Move-all-Equally*, with static competitive ratio in $\Omega(r^2)$ and $2^{O(\sqrt{\log n \cdot \log r})}$ and dynamic competitive ratio in $\Omega(r\sqrt{n})$ and $O(r^{3/2}\sqrt{n})$-competitive. Subsequently, [18] considered MSSC from the viewpoint of online learning. Through dimensionality reduction from permutations to doubly stochastic matrices, they obtained randomized (resp. deterministic) polynomial-time online learning algorithms with $O(1)$-regret for Generalized MSSC (resp. $O(r)$-regret for MSSC).

2 Preliminaries and Basic Definitions

The set of elements e is denoted by U with $|U| = n$. A permutation of the elements is denoted by π where π_i denotes the element lying at position i (for $1 \le i \le n$) and $\mathrm{Pos}(e, \pi)$ denotes the position of the element $e \in U$ in permutation π.

▶ **Definition 1** (Kendall-Tau Distance). *Given the permutations π^A, π^B, a pair of elements (e, e') is inverted if and only if* $\mathrm{Pos}(e, \pi^A) > \mathrm{Pos}(e', \pi^A)$ *and* $\mathrm{Pos}(e, \pi^B) < \mathrm{Pos}(e', \pi^B)$ *or vice versa. The Kendall-Tau distance between the permutations π^A, π^B, denoted by* $\mathrm{d_{KT}}(\pi^A, \pi^B)$*, is the number of inverted pairs.*

▶ **Definition 2** (Spearman' Footrule Distance). *The FootRule distance between the permutations π^A, π^B is defined as* $\mathrm{d_{FR}}(\pi^A, \pi^B) = \sum_{e \in U} |\mathrm{Pos}(e, \pi^A) - \mathrm{Pos}(e, \pi^B)|$.

The Kendall-Tau distance and FootRule distance are approximately equivalent, $\mathrm{d_{KT}}(\pi^A, \pi^B) \le \mathrm{d_{FR}}(\pi^A, \pi^B) \le 2 \cdot \mathrm{d_{KT}}(\pi^A, \pi^B)$. Moreover both of them satisfy the triangle inequality.

▶ **Definition 3.** *An $n \times n$ matrix with positive entries (rows stand for the elements and columns for the positions) is called stochastic if $\sum_{i=1}^{n} A_{ei} = 1$ for all $e \in U$ and doubly stochastic if (additionally) $\sum_{e \in U} A_{ei} = 1$ for all $1 \le i \le n$.*

A permutation of the elements π can be equivalent represented by a 0-1 doubly stochastic matrix A, where $A_{ei} = 1$ if element e lies at position i and 0 otherwise. When clear from context, we use the notion of permutation and (0-1) doubly stochastic matrix interchangeably.

The notion of FootRule distance can be naturally extended to stochastic matrices. Given two doubly stochastic matrices A, B consider the min-cost transportation problem, transforming row A_e to the row B_e where the cost of transporting a unit of mass between column i and column j equals $|i - j|$. Formally for each row e, define a complete bipartite graph where on the left part lie the entries (e, i) for $1 \le i \le n$ and on the right part the entries (e, j) for $1 \le j \le n$. The mass transported from entry (e, i) to entry (e, j) (denoted as f_{ij}^e) costs $f_{ij}^e \cdot |i - j|$ and the total mass *leaving* (e, i) equals A_{ei} and the total mass *arriving* at (e, j) equals B_{ej}.

▶ **Definition 4.** *The FootRule distance between two stochastic matrices A, B, denoted by $\mathrm{d_{FR}}(A, B)$, is the optimal value of the following linear program,*

$$\begin{array}{lll} min & \displaystyle\sum_{e \in U} \sum_{i=1}^{n} \sum_{j=1}^{n} |i - j| \cdot f_{ij}^e & \\ s.t & \displaystyle\sum_{i=1}^{n} f_{ij}^e = B_{ej} & \textit{for all } e \in U \textit{ and } j = 1, \dots, n \\ & \displaystyle\sum_{j=1}^{n} f_{ij}^e = A_{ei} & \textit{for all } e \in U \textit{ and } i = 1, \dots, n \\ & f_{ij}^e \ge 0 & \textit{for all } e \in U \textit{ and } i, j = 1, \dots, n \end{array}$$

▶ **Example 5.** Let the stochastic matrices $A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $B = \begin{pmatrix} 1/3 & 1/3 & 1/3 \\ 1/2 & 1/2 & 0 \\ 1/4 & 0 & 3/4 \end{pmatrix}$. The FootRule distance $\mathrm{d_{FR}}(A, B) = \underbrace{(0 \cdot 1/3 + 1 \cdot 1/3 + 2 \cdot 1/3)}_{\text{first row}} + \underbrace{(1 \cdot 1/2 + 0 \cdot 1/2 + 1 \cdot 0)}_{\text{second row}} + \underbrace{(2 \cdot 1/4 + 1 \cdot 0 + 0 \cdot 3/4)}_{\text{third row}} = 2$.

Up next we present the formal definition of Dynamic Min-Sum Set Cover.

▶ **Definition 6 (Dynamic Min-Sum Set Cover).** *Given a universe of elements U, a sequence of requests $R_1, \dots, R_T \subseteq U$ and an initial permutation of the elements π^0. The goal is to select a sequence of permutation $\pi^1, \dots, \pi^T$ that minimizes*

$$\sum_{t=1}^{T} \pi^t(R_t) + \sum_{t=1}^{T} \mathrm{d_{KT}}(\pi^t, \pi^{t-1})$$

where $\pi^t(R_t)$ is the position of the first element of R_t that we encounter in π^t, $\pi^t(R_t) =$ $\min\{1 \le i \le n : \pi_i^t \in R_t\}$.

We refer to $\sum_{t=1}^{T} \pi^t(R_t)$ as **covering cost** and to $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{KT}}(\pi^t, \pi^{t-1})$ as **moving cost**. We denote with π^t_{Opt} the permutation of the optimal solution of Dyn-MSSC at round t, with o_t the element that the optimal solution uses to cover the request R_t (the element of R_t appearing first in π^t_{Opt}), and with $\mathrm{OPT}_{\text{Dyn-MSSC}}$ the cost of the optimal solution. Finally we call an instance of Dyn-MSSC *r-bounded* in case the cardinality of the requests is bounded by r, $|R_t| \le r$.

3 Approximation Algorithms for Dynamic Min-Sum Set Cover

There exists an approximation-preserving reduction from Set − Cover to Dyn-MSSC that provides us with the following inapproximability results.

▶ **Theorem 7.**
- *There is no $c \cdot \log n$-approximation algorithm for* Dyn-MSSC *(for a sufficienly small constant c) unless* P = NP*.*
- *For r-bounded sequences, there is no $o(r)$-approximation algorithm for* Dyn-MSSC*, unless there is a $o(r)$-approximation algorithm for* Set − Cover *with each element being covered by at most r sets.*

The proof of Theorem 7 is fairly simple, given an instance of Set − Cover we construct an instance of Dyn-MSSC in which the initial permutation π^0 contains in the first positions some dummy elements (they do not appear in any of the requests) and in the last positions the sets of the Set − Cover (we consider an element of Dyn-MSSC for each set of Set − Cover). Finally each request for Dyn-MSSC is associated with an element of the Set − Cover and contains the *elements in* Dyn-MSSC/ *sets in* Set − Cover containing it.

Both the $O(\log^2 n)$-approximation algorithm (for requests of general cardinality) and the $O(r^2)$-approximation algorithm for r-bounded requests, that we subsequently present, are based on rounding a linear program called *Fractional Move To Front*. The latter is the linear program relaxation of *Move To Front*, a problem closely related to Dynamic Min-Sum Set Cover. MTF asks for a sequence of permutations $\pi^1, \ldots, \pi^T$ such as at each round t, an element of R_t lies on the first position of π^t and $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(\pi^t, \pi^{t-1})$ is minimized.

▶ **Definition 8.** *Given a sequence of requests $R_1, \ldots, R_T \subseteq U$ and an initial permutation of the elements π^0, consider the following linear program, called* Fractional − MTF*,*

$$
\begin{aligned}
\textit{min} \quad & \sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(A^t, A^{t-1}) \\
\textit{s.t} \quad & \sum_{i=1}^{n} A^t_{ei} = 1 \qquad \textit{for all } e \in U \textit{ and } t = 1, \ldots, T \\
& \sum_{e \in U} A^t_{ei} = 1 \qquad \textit{for all } i = 1, \ldots, n \textit{ and } t = 1, \ldots, T \\
& \sum_{e \in R_t} A^t_{e1} = 1 \qquad \textit{for all } t = 1, \ldots, T \\
& A^0 = \pi^0 \\
& A^t_{ei} \ge 0 \qquad \textit{for all } e \in U,\ i = 1, \ldots, n \textit{ and } t = 1, \ldots, T
\end{aligned}
$$

where $\mathrm{d}_{\mathrm{FR}}(\cdot, \cdot)$ is the FootRule distance of Definition 4.

There is an elegant argument (appeared in previous works, e.g., [17]) showing that the optimal solution of MTF is at most $4 \cdot \mathrm{OPT}_{\text{Dyn-MSSC}}$. In Lemma 9 we provide the argument (see Appendix A) and establish that Fractional − MoveToFront is a 4-approximate relaxation of Dyn-MSSC.

▶ **Lemma 9.** $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(A^t, A^{t-1}) \leq 4 \cdot \mathrm{OPT}_{\text{Dyn-MSSC}}$ *where* $A^1, \ldots, A^t$ *is the optimal solution of* Fractional – MTF.

As already mentioned, our main technical contribution is the design of *rounding schemes* converting the optimal solution, $A^1, \ldots, A^T$, of Fractional – MTF into a sequence of permutations $\pi^1, \ldots, \pi^T$. This is done so as to bound the moving cost of our algorithms by the moving cost $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(A^t, A^{t-1})$. We then separately bound the covering cost, $\sum_{t=1}^{T} \pi^t(R_t)$ by showing that always an element of R_t lies on the first positions of π^t.

The main technical challenge in the design of our rounding schemes is ensure to that the moving cost of our solutions $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{KT}}(\pi^t, \pi^{t-1})$ is approximately bounded by the moving cost $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(A^t, A^{t-1})$. Despite the fact that the connection between doubly stochastic matrices and permutations is quite well-studied and there are various rounding schemes converting doubly stochastic matrices to probability distributions on permutations (such as the Birkhoff–von Neumann decomposition or the schemes of [8, 25, 6, 17]), using such schemes in a *black-box* manner does not provide any kind of positive results for Dyn-MSSC. For example consider the case where $A^1 = \cdots = A^T$ and thus $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(A^t, A^{t-1}) = \mathrm{d}_{\mathrm{FR}}(A^1, A^0)$. In case a randomized rounding scheme is applied *independently to each* A^t, there always exists a positive probability that $\pi^t \neq \pi^{t-1}$ and thus the moving cost will far exceed $\mathrm{d}_{\mathrm{FR}}(A^1, A^0)$ as T grows. The latter reveals the need for *coupled rounding schemes* that convert the overall sequence of matrices $A^1, \ldots, A^T$ to a sequence of permutations $\pi^1, \ldots, \pi^T$. Such a rounding scheme is presented in Algorithm 1 and constitutes the back-bone of our approximation algorithm for requests of general cardinality.

Algorithm 1 A Randomized Algorithm for Dyn-MSSC

Input: A sequence of requests $R_1, \ldots, R_T$ and an initial permutation of the elmenents π^0.
Output: A sequence of permutations $\pi^1, \ldots, \pi^T$.

```
1: Find the optimal solution A^0 = π^0, A^1, ..., A^T for Fractional − MTF.
2: for each element e ∈ U do
3:     Select α_e uniformly at random in [0, 1].
4: end for
5: for t = 1 ... T do
6:     for all elements e ∈ U do
7:         I_e^t := argmin_{1≤i≤n} {log n · Σ_{s=1}^{i} A_{es}^t ≥ α_e}.
8:     end for
9:     π^t := sort elements according to I_e^t with ties being broken lexicographically.
10: end for
```

The rounding scheme described in Algorithm 1, imposes correlation between the different time-steps by simply requiring that each element e selects α_e once and for all and by breaking ties lexicographically (any consistent tie-breaking rule would also work). In Lemma 12 of Section 4, we show that no matter the sequence of doubly stochastic matrices, the rounding scheme of Algorithm 1 produces a sequence of permutations with overall moving cost at most $4\log^2 n$ the moving cost of the matrix-sequence[1] and thus establishes that the overall moving cost of Algorithm 1 is bounded by $4 \log^2 n \cdot \mathrm{OPT}_{\text{Dyn-MSSC}}$. The $\log n$ multiplication in Step 7 serves as a *probability amplifier* ensuring that at least one element of R_t lies in

[1] By omitting the $\log n$-multiplication step of Step 7, one could establish that the moving cost of the produced permutations is at most 4 times the moving cost of the matrix-sequence, however omitting the $\log n$ multiplication could lead in prohibitively high covering cost.

the relatively first positions of π^t and permits us to approximately bound the covering cost $\sum_{t=1}^T \mathbb{E}\left[\pi^t(R_t)\right]$ by the covering cost of the optimal solution for Dyn-MSSC, $\sum_{t=1}^T \pi^t_{\text{Opt}}(R_t)$.

▶ **Theorem 10.** *Algorithm 1 is a $O(\log^2 n)$-approximation algorithm for* Dyn-MSSC*.*

Despite the fact that in Step 7 of Algorithm 1, we multiply the entries of A^t with $\log n$ the overall guarantee is $O(\log^2 n)$. At a first glance the latter seems quite strange but admits a rather natural explanation. For most of the positions i, the probability that an element e admits index $I_e^t = i$ is roughly $\log n \cdot A^t_{ei}$, but due to the fact each index $j \leq i$ is on expectation selected by $\log n$ other elements, the expected position of e in the produced permutation is roughly $\log^2 n$ times the expected value of $\text{argmin}_{1 \leq i \leq n}\{\sum_{s=1}^i A^t_{es} \geq \alpha_e\}$. This phenomenon relates with the elegant fitting argument given in [15] to prove that the greedy algorithm is 4-approximation for the original *Min-Sum Set Cover* (which is tight unless P = NP). The latter makes us conjecture that the tight inapproximability bound for Dyn-MSSC is $\Omega(\log^2 n)$ for requests of general cardinality.

Motivated by the r-approximation LP-based algorithm for instances of $\text{Set} - \text{Cover}$ in which elements belong in at most r sets, we examine whether the $O(\log^2 n)$ for Dyn-MSSC can be ameliorated in case of r-bounded request sequences. Interestingly, the simple *greedy rounding* scheme (described[2] in Algorithm 2) provides such a $O(r^2)$-approximation algorithm.

Algorithm 2 A Greedy-Rounding Algorithm for Dyn-MSSC for r-Bounded Sequences.

Input: A request sequence $R_1, \ldots, R_T$ with $|R_t| \leq r$ and an initial permutation π^0.
Output: A sequence of permutations $\pi^1, \ldots, \pi^T$.

1: Find the optimal solution $A^0 = \pi^0, A^1, \ldots, A^T$ for $\text{Fractional} - \text{MTF}$.
2: **for** $t = 1 \ldots T$ **do**
3: $\quad \pi^t :=$ in π^{t-1}, move to the first position an element $e \in R_t$ such that $A^t_{e1} \geq 1/r$
4: **end for**

The $O(r^2)$-approximation guarantee of Algorithm 3 is formally stated and proven in Theorem 11. The main technical challenge is that we cannot directly compare the moving cost of Algorithm 2 with $\sum_{t=1}^T \text{d}_{\text{FR}}(A^t, A^{t-1})$ and thus we deploy a two-step detour.

In the first step (Lemma 19), we prove the existence of a sequence of doubly stochastic matrices $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ for which each $\hat{A}^t$ satisfies that **(i)** its entries of are **multiples of** $1/r$, **(ii)** $\hat{A}^t_{e_t 1} \geq 1/r$ where e_t is the element that Algorithm 2 moves to the first position at round t, and **(iii)** the sequence $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ admits moving cost at most $\sum_{t=1}^T \text{d}_{\text{FR}}(A^t, A^{t-1})$. In order to establish the existence of such a sequence, we construct an appropriate linear program (see Definition 18) based on the elements that Algorithm 2 moves to the first position at each round and prove that it admits an optimal solution with values being multiples of $1/r$. To do the latter, we relate the linear program of Definition 18 with a fractional version of the k-Paging [7] problem and based on the optimal eviction policy (*evict the page appearing the furthest in the future*), we design an algorithm producing optimal solutions for the LP with values being multiple of $1/r$.

In the second step (Lemma 20), we show that for any sequence $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ satisfying properties **(i)** and **(ii)**, the moving cost of Algorithm 2 is at most $O(r^2) \cdot \sum_{t=1}^T \text{d}_{\text{FR}}(\hat{A}^t, \hat{A}^{t-1})$. The latter is achieved through the use of an appropriate *potential*

[2] Step 3 of Algorithm 2 is well-defined since $|R_t| \leq r$ and $\sum_{e \in R_t} A^t_{e1} = 1$.

function based on a generalization of Kendall-Tau distance to doubly stochastic matrices with entries being multiples of $1/r$ (see Definition 22).

▶ **Theorem 11.** *Algorithm 2 is a $O(r^2)$-approximation algorithm for* Dyn-MSSC.

In Section 4 and 5 we provide the basic steps and ideas in the proof of Theorem 10 and 11 respectively.

4 Proof of Theorem 10

The basic step towards the proof of Theorem 10 is Lemma 12, establishing the fact that once two doubly stochastic matrices are given as input to the randomized rounding of Algorithm 1, the expected distance of the produced permutations is approximately bounded by the distance of the respective doubly stochastic matrices.

▶ **Lemma 12.** *Let the doubly stochastic matrices A, B given as input to the rounding scheme of Algorithm 1. Then for the produced permutations π^A, π^B, $\mathbb{E}\left[\mathrm{d}_{\mathrm{KT}}(\pi^A, \pi^B)\right] \leq 4\log^2 n \cdot \mathrm{d}_{\mathrm{FR}}(A, B)$.*

Before exhibiting the proof of Lemma 12 we introduce the notion of *neighboring matrices*.

▶ **Definition 13.** *(Neighboring stochastic matrices) The stochastic matrices A, B are neighboring if and only if they differ in exactly two entries lying on the same row and on consecutive columns.*

▶ **Example 14.** Let $A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $B = \begin{pmatrix} 1/2 & 1/2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ and $C = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. The pair of matrices (A, B) and (A, C) are neighboring while (B, C) are not.

Any doubly stochastic matrix A can be converted to another doubly stochastic matrix B through an intermediate sequence of neighboring stochastic matrices all of which are *almost doubly stochastic* and their overall moving cost equals $\mathrm{d}_{\mathrm{FR}}(A, B)$.

▷ Claim 15. Given the doubly stochastic matrices A, B, there exists a finite sequence of stochastic matrices, $A^0, \ldots, A^T$ such that

1. $A^0 = A$ and $A^T = B$.
2. A^t and A^{t-1} are neighboring.
3. the column-sum is bounded by 2, $\sum_{e \in U} A^t_{ei} \leq 2$ for all $1 \leq i \leq n$.
4. $\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(A^t, A^{t-1}) = \mathrm{d}_{\mathrm{FR}}(A, B)$.

▶ **Example 16.** Let the doubly stochastic matrices $A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $B = \begin{pmatrix} 0 & 0 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 1/2 & 0 \end{pmatrix}$. A can be converted to B with the following sequence neighboring stochastic matrices, $\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$, $\begin{pmatrix} 0 & 0 & 1 \\ 1/2 & 1/2 & 0 \\ 0 & 1 & 0 \end{pmatrix}$, $\begin{pmatrix} 0 & 0 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 1/2 & 0 \end{pmatrix}$. Notice that the above sequence satisfies all the 4 requirements of Claim 15.

The notion of neighboring matrices is rather helpful since Lemma 12 admits a fairly simple proof in case A, B are neighboring stochastic matrices (notice that the rounding scheme of Algorithm 1 is well-defined even for stochastic matrices). The latter is formally stated and proven in Lemma 17 and is the main technical claim of the section.

▶ **Lemma 17.** *Let π^A, π^B the permutations produced by the rounding scheme of Algorithm 1 (given as input) the stochastic matrices A, B that **i)** are neighboring **ii)** their column-sum is bounded by 2, then $\mathbb{E}[\mathrm{d_{KT}}(\pi^A, \pi^B)] \leq 4\log^2 n \cdot \mathrm{d_{FR}}(A, B)$*

The proof of Lemma 12 easily follows by Claim 15 and Lemma 17 (see Appendix B). We conclude the section with the proof of Theorem 10.

Proof of Theorem 10. By Lemma 12 and Lemma 9,

$$\sum_{t=1}^{T} \mathbb{E}\left[\mathrm{d_{KT}}(\pi^t, \pi^{t-1})\right] \leq 4\log^2 n \cdot \sum_{t=1}^{T} \mathrm{d_{FR}}(A^t, A^{t-1}) \leq 4\log^2 n \cdot \mathrm{OPT}_{\text{Dyn-MSSC}}$$

Up next we bound the expected covering cost $\sum_{t=1}^{T} \mathbb{E}\left[\pi^t(R_t)\right]$. Notice that since $\sum_{e \in R_t} A^t_{e1} = 1$, the only elements that can have index $I^t_e = 1$ are the elements $e \in R_t$. As a result, in case there exists some e at round t with $I^t_e = 1$ then $\pi^t(R_t) = 1$.

$$\begin{aligned}
\mathbb{E}\left[\pi^t(R_t)\right] &\leq 1 + n \cdot \Pr\left[I^t_e > 1 \text{ for all } e \in R_t\right] \\
&\leq 1 + n \cdot \Pi_{e \in R_t}\left(1 - \log n \cdot A^t_{e1}\right) \\
&\leq 1 + n \cdot e^{-\log n \cdot \sum_{e \in R_t} A^t_{e1}} = 2 \cdot \pi^t_{\mathrm{Opt}}(R_t)
\end{aligned}$$

where the last inequality follows due to the fact that $\sum_{e \in R_t} A^t_{e1} = 1$ and $\pi^t_{\mathrm{Opt}}(R_t) \geq 1$. ◀

5 Proof of Theorem 11

In this section we present the basic steps towards the proof of Theorem 11. We remind that $|R_t| \leq r$ and we denote with e_t the element that Algorithm 2 moves in the fist position at round t. As already mentioned, the proof is structured in two different steps.

1. We prove the existence of a sequence of doubly stochastic matrices $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ such that **(i)** the entries of each $\hat{A}^t$ are multiples of $1/r$, **(ii)** each $\hat{A}^t$ admits $1/r$ mass for element e_t in first position ($\hat{A}^t_{e_t 1} \geq 1/r$) and **(iii)** $\sum_{t=1}^{T} \mathrm{d_{FR}}(\hat{A}^t, \hat{A}^{t-1}) \leq \sum_{t=1}^{T} \mathrm{d_{FR}}(A^t, A^{t-1})$.
2. We use properties **(i)** and **(ii)** to prove that the moving cost of Algorithm 2 is roughly upper bounded by $\Theta(r^2) \cdot \sum_{t=1}^{T} \mathrm{d_{FR}}(\hat{A}^t, \hat{A}^{t-1})$.

We start with the construction of the sequence $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$.

▶ **Definition 18.** *For the sequence of elements $e_1, \ldots, e_T \in U$ (the elements that Algorithm 2 moves to the fist position at each round), consider the following linear program,*

$$\begin{array}{lll}
\mathit{min} & \displaystyle\sum_{t=1}^{T} \mathrm{d_{FR}}(\hat{A}^t, \hat{A}^{t-1}) & \\
\mathit{s.t} & \displaystyle\sum_{i=1}^{n} \hat{A}^t_{ei} = 1 & \textit{for all } e \in U \textit{ and } t = 1, \ldots, T \\
& \displaystyle\sum_{e \in U} \hat{A}^t_{ei} = 1 & \textit{for all } i = 1, \ldots, n \textit{ and } t = 1, \ldots, T \\
& \hat{A}^t_{e_t 1} \geq 1/r & \textit{for all } t = 1, \ldots, T \\
& \hat{A}^0 = \pi^0 & \\
& \hat{A}^t_{ei} \geq 0 & \textit{for all } e \in U,\ i = 1, \ldots, n \textit{ and } t = 1, \ldots, T
\end{array}$$

The sequence $\hat{A}^0 = \pi^0, \ldots, \hat{A}^T$ is defined as the optimal solution of the LP in Definition 18 with the entries of each $\hat{A}^t$ being **multiples of** $1/r$. The existence of such an optimal solution is established in Lemma 19.

▶ **Lemma 19.** *There exists an optimal solution $\hat{A} = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ for the linear program of Definition 19 such that entries of each $\hat{A}^t$ are multiples of $1/r$.*

The proof of Lemma 19 is one of the main technical contributions of this work. Due to lack of space its proof is deferred to Appendix C. We remark that the *semi-integrality property*, that Lemma 19 states, is not due to the properties of the LP's polytope and in fact there are simple instances in which the optimal extreme points do not satisfy it. We establish Lemma 19 via the design of an optimal algorithm for the LP of Definition 18 (Algorithm 3) that always produces solutions with entries being multiples of $1/r$. Up next we describe in brief the idea behind Algorithm 3.

Given the matrix $\hat{A}^{t-1}$, Algorithm 3 construct $\hat{A}^t$ as follows. At first it moves $1/r$ mass from the left-most entry (e_t, j) with $\hat{A}^{t-1}_{e_t j} \geq 1/r$ to the entry $(e_t, 1)$. At this point the third constraint of the LP in Definition 18 is satisfied but the column-stochasticity constraints are violated (the first column admits mass $1 + 1/r$ and the j-th column admits mass $1 - 1/r$). Algorithm 3 inductively restores column-stochasticity from left to right. At step i, all the columns on the left of i are restored and the violations concern the column i and j (i's mass is $1 + 1/r$ and j's mass is $1 - 1/r$). Now Algorithm 3 must move a total of $1/r$ mass from column i to column $i + 1$. In case there exists an element e with total amount of mass greater than $2/r$, Algorithm 2 moves the $1/r$ mass from the entry (e, i) to the entry $(e, i + 1)$. The reason is that even if $e = e_{t'}$ at some future round t', the third constraint only requires $1/r$ mass. In case there is no such element, Algorithm 3 moves the $1/r$ mass from the element appearing the furthest in the sequence $\{e_t, \ldots, e_T\}$. The latter is in accordance with the optimal eviction policy for $\mathrm{k-Paging}$ which at each round evicts the page appearing furthest in the future [7]. The optimality of Algorithm 3 is established in Lemma 30 of Appendix C and the fact that produced solution admits values being $1/r$ is inductively established.

To this end, we can show that all of the desired properties of the sequence $\hat{A} = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ are satisfied. Property **(i)** is established by Lemma 19. Property **(ii)** is enforced by the constraint $\hat{A}^t_{e_t 1} \geq 1/r$. Now for Property **(iii)**, notice that by the definition of Algorithm 2, $A^t_{e_t 1} \geq 1/r$. As a result, the sequence $A^0 = \pi^0, A^1, \ldots, A^T$ is feasible for the linear program of Definition 18 and thus $\sum_{t=1}^T \mathrm{d}_{\mathrm{FR}}(\hat{A}^t, \hat{A}^{t-1}) \leq \sum_{t=1}^T \mathrm{d}_{\mathrm{FR}}(A^t, A^{t-1})$.

▶ **Lemma 20.** *Let $\pi^0, \pi^1, \ldots, \pi^T$ the permutations produced by Algorithm 2 and $e_1, \ldots, e_T$ the elements that Algorithm 2 moves to the first position at each round t. For any sequence of doubly stochastic matrices $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ for which Property **(i)** and Property **(ii)** are satisfied, $\sum_{t=1}^T \mathrm{d}_{\mathrm{KT}}(\pi^t, \pi^{t-1}) \leq 2r^2 \cdot \sum_{t=1}^T \mathrm{d}_{\mathrm{FR}}(\hat{A}^t, \hat{A}^{t-1}) + r \cdot T$.*

The proof of Theorem 11 directly follows by Lemma 19 and 20. In Section 5.1 we present the basic steps for of Lemma 19.

5.1 Proof of Lemma 20

In order to prove Lemma 20, we make use of an appropriate potential function that can be viewed as an extension of the Kendall-Tau distance (see Definition 2) to doubly stochastic matrices with entries being multiples of $1/r$.

▶ **Definition 21** (r-**Index**)**.** *The r-index of an element $e \in U$ in the doubly stochastic matrix A, $I_e^A := \operatorname{argmin}\{1 \leq i \leq n : \sum_{s=1}^i A_{es} \geq 1/r\}$*

▶ **Definition 22 (Fractional Kendall-Tau Distance).** *Given the doubly stochastic matrices A, B, a pair of elements $(e, e') \in U \times U$ is inverted if and only if one of the following condition holds,*

1. $I^A_e > I^A_{e'}$ *and* $I^B_e < I^B_{e'}$.
2. $I^A_e < I^A_{e'}$ *and* $I^B_e > I^B_{e'}$.
3. $I^A_e = I^A_{e'}$ *and* $I^B_e \neq I^B_{e'}$.
4. $I^A_e \neq I^A_{e'}$ *and* $I^B_e = I^B_{e'}$.

The fractional Kendall-Tau distance between two doubly stochastic matrices A, B, denoted as $\mathrm{d_{KT}}(A, B)$, is the number of inverted pairs of elements.

Notice that in case of $0-1$ doubly stochastic matrices the Fractional Kendall-Tau distance of Definition 22 coincides with the Kendall-Tau distance of Definition 2.

▷ Claim 23. Fractional Kendall-Tau Distance satisfies the triangle inequality, $\mathrm{d_{KT}}(A, B) \leq \mathrm{d_{KT}}(A, C) + \mathrm{d_{KT}}(C, B)$.

In the case of doubly stochastic matrices with their entries being multiples of $1/r$, Fractional Kendall-Tau distance relates to FootRule distance of Definition 4.

▶ **Lemma 24.** *Let the doubly stochastic matrices A, B with entries that are multiples of $1/r$. Then $\mathrm{d_{KT}}(A, B) \leq 2r^2 \cdot \mathrm{d_{FR}}(A, B)$.*

We conclude the section with Lemma 25. Then Lemma 20 follows by Lemma 25 and 24.

▶ **Lemma 25.** *Let $\pi^0, \pi^1, \ldots, \pi^T$ the permutations produced by Algorithm 2 and $e_1, \ldots, e_T$ the elements that Algorithm 2 moves to the first position at each round t. For any sequence of doubly stochastic matrices $B^0 = \pi^0, B^1, \ldots, B^T$ with $B^t_{e_t 1} \geq 1/r$,*

$$\sum_{t=1}^{T} \mathrm{d_{KT}}(\pi^t, \pi^{t-1}) \leq \sum_{t=1}^{T} \mathrm{d_{KT}}(B^t, B^{t-1}) + r \cdot T$$

The proof of Lemma 25 is based on the following two inequalities, $\mathrm{d_{KT}}(\pi^t, \pi^{t-1}) + \mathrm{d_{KT}}(\pi^t, B^t) - \mathrm{d_{KT}}(\pi^{t-1}, B^t) \leq r$ and $\mathrm{d_{KT}}(\pi^{t-1}, B^t) - \mathrm{d_{KT}}(\pi^{t-1}, B^{t-1}) \leq \mathrm{d_{KT}}(B^t, B^{t-1})$. The second inequality follows by the triangle inequality established in Claim 23. The first follows by the fact that $I^{B^t}_{e_t} = 1$ and the definition of Fractional Kendall-Tau distance (see Appendix C).

6 Concluding Remarks

In this work we examine the polynomial-time approximability of Dynamic Min-Sum Set Cover. We present $\Omega(\log n)$ and $\Omega(r)$ inapproximability results for general and r-bounded request sequences, while we respectively provide $O(\log^2 n)$ and $O(r^2)$ polynomial-time approximation algorithms. Closing this gap is an interesting question that our work leaves open. Another interesting research direction concerns the competitive ratio in the online version of Dynamic Min-Sum Set Cover. [18] provides an $\Omega(r)$ lower bound and a $\Theta\left(r^{3/2}\sqrt{n}\right)$-competitive online algorithm for r-bounded sequences. Designing online algorithms for a relaxation of the problem (such as the Fractional – MTF) and using the rounding schemes that this work suggests may be a fruitful approach towards closing this gap.

References

1. Noga Alon, Dana Moshkovitz, and Shmuel Safra. Algorithmic construction of sets for k-restrictions. *ACM Transactions on Algorithms (TALG)*, 2(2):153–177, 2006.
2. Christoph Ambühl. Offline list update is NP-hard. In *Algorithms - ESA 2000, 8th Annual European Symposium, Proceedings*, volume 1879 of *Lecture Notes in Computer Science*, pages 42–51. Springer, 2000.
3. Hyung-Chan An, Ashkan Norouzi-Fard, and Ola Svensson. Dynamic facility location via exponential clocks. *ACM Trans. Algorithms*, 13(2):21:1–21:20, 2017.
4. Yossi Azar and Iftah Gamzu. Ranking with submodular valuations. In *SODA*, pages 1070–1079, 2011.
5. Yossi Azar, Iftah Gamzu, and Xiaoxin Yin. Multiple intents re-ranking. In *STOC*, pages 669–678, 2009.
6. Nikhil Bansal, Jatin Batra, Majid Farhadi, and Prasad Tetali. Improved approximations for min sum vertex cover and generalized min sum set cover. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021*, pages 998–1005. SIAM, 2021.
7. Nikhil Bansal, Niv Buchbinder, and Joseph Naor. A primal-dual randomized algorithm for weighted paging. *J. ACM*, 59(4):19:1–19:24, 2012.
8. Nikhil Bansal, Anupam Gupta, and Ravishankar Krishnaswamy. A constant factor approximation algorithm for generalized min-sum set cover. In *SODA*, pages 1539–1545, 2010.
9. Amotz Bar-Noy, Mihir Bellare, Magnús M. Halldórsson, Hadas Shachnai, and Tami Tamir. On chromatic sums and distributed resource allocation. *Inf. Comput.*, 140(2):183–202, 1998.
10. Omer Ben-Porat and Moshe Tennenholtz. A game-theoretic approach to recommendation systems with strategic content providers. In *Annual Conference on Neural Information Processing Systems 2018*, NeurIPS 2018, 2018.
11. Guillaume Cabanac and Thomas Preuss. Capitalizing on order effects in the bids of peer-reviewed conferences to secure reviews by expert referees. *J. Am. Soc. Inf. Sci. Technol.*, 64(2):405–415, February 2013. doi:10.1002/asi.22747.
12. Mahsa Derakhshan, Negin Golrezaei, Vahideh Manshadi, and Vahab Mirrokni. Product ranking on online platforms. In *Proc. of the 21st ACM Conference on Economics and Computation (EC 2015)*. ACM, 2020. URL: https://ssrn.com/abstract=3130378.
13. Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 613–622, New York, NY, USA, 2001. Association for Computing Machinery.
14. David Eisenstat, Claire Mathieu, and Nicolas Schabanel. Facility location in evolving metrics. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 459–470. Springer, 2014.
15. Uriel Feige, László Lovász, and Prasad Tetali. Approximating min sum set cover. *Algorithmica*, 40(4):219–234, 2004.
16. Tanner Fiez, Nihar Shah, and Lillian Ratliff. A super* algorithm to determine orderings of items to show users. In *Conference on Uncertainty in Artificial Intelligence*, 2020.
17. Dimitris Fotakis, Loukas Kavouras, Grigorios Koumoutsos, Stratis Skoulakis, and Manolis Vardas. The online min-sum set cover problem. In *Proc. of the 47th International Colloquium on Automata, Languages and Programming (ICALP 2020)*, LIPIcs, 2020.
18. Dimitris Fotakis, Thanasis Lianeas, Georgios Piliouras, and Stratis Skoulakis. Efficient online learning of optimal rankings: Dimensionality reduction via gradient descent. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.
19. Anupam Gupta, Kunal Talwar, and Udi Wieder. Changing bases: Multistage optimization for matroids and matchings. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 563–575. Springer, 2014.

20 Refael Hassin and Asaf Levin. An approximation algorithm for the minimum latency set cover problem. In *ESA*, pages 726–733, 2005.

21 Sungjin Im. Min-sum set cover and its generalizations. In *Encyclopedia of Algorithms*, pages 1331–1334. Springer, 2016.

22 Sungjin Im, Viswanath Nagarajan, and Ruben van der Zwaan. Minimum latency submodular cover. *ACM Trans. Algorithms*, 13(1):13:1–13:28, 2016.

23 Sungjin Im, Maxim Sviridenko, and Ruben van der Zwaan. Preemptive and non-preemptive generalized min sum set cover. *Math. Program.*, 145(1-2):377–401, 2014.

24 Alejandro López-Ortiz, Marc P. Renault, and Adi Rosén. Paid exchanges are worth the price. *Theoretical Computer Science*, 824-825:1–10, 2020.

25 Martin Skutella and David P. Williamson. A note on the generalized min-sum set cover problem. *Oper. Res. Lett.*, 39(6):433–436, 2011.

26 Daniel Dominic Sleator and Robert Endre Tarjan. Self-adjusting binary search trees. *J. ACM*, 32(3):652–686, 1985.

27 Matthew J. Streeter, Daniel Golovin, and Andreas Krause. Online learning of assignments. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009*, pages 1794–1802. Curran Associates, Inc., 2009.

28 Erez Timnat. The list update problem, 2016. Master Thesis, Technion- Israel Institute of Technology.

A Omitted Proofs of Section 3

Proof. Let the equivalent definition of Set − Cover in which we are given a universe of element $E = \{1, \ldots, n\}$ and sets $S_1, S_2, \ldots, S_m \subseteq E$ and we are asked to select the minimum number of elements covering all the sets (an element e covers set S_i if $e \in S_i$).

Consider the instance of Dyn-MSSC with the elements $U = \{1, \ldots, n\} \cup \{d_1, \ldots, d_{n^2m}\}$. The elements $\{d_1, \ldots, d_{n^2m}\}$ are dummy in the sense that they appear in none of the requests R_t. Let the initial permutation π_0 contain in the first n^2m positions the dummy elements and in the last n positions the elements $\{1, \ldots, n\}$, $\pi_0 = [d_1, \ldots, d_{n^2m}, 1, \ldots, n]$ and the request sequence of Dyn-MSSC be $S_1, S_2, \ldots, S_m$.

Let a c-approximation algorithm for Dyn-MSSC producing the permutation $\pi_1, \ldots, \pi_m$ the cost of which is denoted by Alg. Let also CoverAlg denote the set composed by the element that the c-approximation algorithm uses to cover the requests, CoverAlg $=$ {the element of S_t appearing first in π_t}. Then,

$$\text{Alg} \geq n^2 m \cdot |\text{CoverAlg}|$$

Now consider the following solution for Dyn-MSSC constructed by the optimal solution for Set − Cover. This solution initially moves the elements of the optimal covering set $\text{OPT}_{\text{SetCover}}$ to the first positions and then never changes the permutation. Clearly the cost of this solution is upper bounded by

$$\text{Set} - \text{Cover}_{\text{Dyn-MSSC}} \leq \underbrace{|\text{OPT}_{\text{SetCover}}| \cdot (n^2 m + n)}_{\text{moving cost}} + \underbrace{m \cdot |\text{OPT}_{\text{SetCover}}|}_{\text{covering cost}}$$

In case $\text{Alg} \leq c \cdot \text{Set} - \text{Cover}_{\text{Dyn-MSSC}}$, we directly get that $|\text{CoverAlg}| \leq 3c \cdot |\text{OPT}_{\text{SetCover}}|$.

There is no polynomial-time approximation algorithm for Set−Cover with approximation ratio better than $\log m$. The latter holds even for instance of Set − Cover for which $m = \text{poly}(n)$ [1] where $\text{poly}(\cdot)$ is a polynomial with degree bounded by a universal constant. Since the number of elements $|U|$, in the constructed instance of Dyn-MSSC is n^2m, any $c \cdot \log |U|$-approximation for Dyn-MSSC (for c sufficiently small) implies an approximation algorithm for Set − Cover with approximation ratio less than $\log n$. In case there exists an $c = o(r)$-approximation algorithm for Dyn-MSSC for requests sequences $R_1, \ldots, R_T$ where $|R_t| \leq r$, we obtain an $o(r)$-approximation for algorithm for Set − Cover for sets with cardinality bounded by r. In the standard form of Set − Cover this is translated into the fact that each element belongs in at most r sets. ◀

Proof of Lemma 9. Let o_t the element of R_t appearing first in the permutation π^t_{Opt}. Consider the sequence of permutation $\pi^0, \pi^1, \ldots, \pi^T$ constructed by moving at each round t, the element o_t to the first position of the permutation. Notice that $\pi^0, \pi^1, \ldots, \pi^T$ is a feasible solution for both MoveToFront and Fractional − MTF. The first key step towards the proof of Lemma 9 is that

$$\text{d}_{\text{KT}}(\pi^t, \pi^{t-1}) + \text{d}_{\text{KT}}(\pi^t, \pi^t_{\text{Opt}}) - \text{d}_{\text{KT}}(\pi^{t-1}, \pi^t_{\text{Opt}}) \leq 2 \cdot \pi^t_{\text{Opt}}(R_t)$$

To understand the above inequality, let k_t be the position of o_t in permutation π^{t-1}. Out of the $k_t - 1$ elements on the right of o_t in permutation π^{t-1}, let $Left_t$ ($Right_t$) denote the elements that are on the left (right) of o_t in permutation π^{t-1}_{Opt}. It is not hard to see that $\pi^t_{\text{Opt}}(R_t) \geq |Left_t|$, $\text{d}_{\text{KT}}(\pi^t, \pi^{t-1}) = |Left_t| + |Right_t|$ and $\text{d}_{\text{KT}}(\pi^t, \pi^t_{\text{Opt}}) - \text{d}_{\text{KT}}(\pi^{t-1}, \pi^t_{\text{Opt}}) = |Left_t| -$

$|Right_t|$. Using the fact that $\mathrm{d_{KT}}(\pi^t, \pi^t_{\mathrm{Opt}}) - \mathrm{d_{KT}}(\pi^{t-1}, \pi^t_{\mathrm{Opt}}) \leq \mathrm{d_{KT}}(\pi^t_{\mathrm{Opt}}, \pi^{t-1}_{\mathrm{Opt}})$ and the previous inequality we get,

$$\mathrm{d_{KT}}(\pi^t, \pi^{t-1}) + \mathrm{d_{KT}}(\pi^t, \pi^t_{\mathrm{Opt}}) - \mathrm{d_{KT}}(\pi^{t-1}, \pi^{t-1}_{\mathrm{Opt}}) \leq 2 \cdot \pi^t_{\mathrm{Opt}}(R_t) + \mathrm{d_{KT}}(\pi^t_{\mathrm{Opt}}, \pi^{t-1}_{\mathrm{Opt}})$$

and by a telescopic sum we get $\sum_{t=1}^{T} \mathrm{d_{KT}}(\pi^t, \pi^{t-1}) \leq 2 \cdot \mathrm{OPT}_{\text{Dyn-MSSC}}$. The proof follows by the fact that $\mathrm{d_{FR}}(\pi^t, \pi^{t-1}) \leq 2 \cdot \mathrm{d_{KT}}(\pi^t, \pi^{t-1})$. ◀

B Omitted Proofs of Section 4

Proof Sketch of Claim 15. Let f^e_{ij} denotes the optimal solution of the linear program of Definition 4 defining the FootRule distance $\mathrm{d_{FR}}(A, B)$. In case $A \neq B$, there exist elements e_1, e_2 and indices $i < j$ such that $f^{e_1}_{i\ell(i)} > 0$ and $f^{e_2}_{j\ell(j)} > 0$ with $\ell(i) >= j$ and $\ell(j) <= i$.

Let $\epsilon = \min(f^{e_1}_{i\ell(i)}, f^{e_2}_{j\ell(j)})$ and consider the sequence of the $|i-j|$ matrices produced by moving ϵ amount of mass in row e_1 from column i to column j. Then consider the sequence of the $|i-j|$ matrices produced by moving ϵ amount of mass in the row e_2 from column j to column i.

In the overall sequence of $2|i-j|$ stochastic matrices, two consecutive matrices are *neighboring*. Furthermore the column-sum of the matrices does not exceed $1 + \epsilon \leq 2$ and the final matrix A' of the sequence is doubly stochastic. Moreover by the fact that $t(i) \geq j$ and $t(j) \leq i$ we get that the overall moving cost of the sequence equals $\mathrm{d_{FR}}(A, A')$ and that $\mathrm{d_{FR}}(A, B) = \mathrm{d_{FR}}(A, A') + \mathrm{d_{FR}}(A', B)$. Applying the same argument inductively, until we reach matrix B, proves Claim 15. ◀

Proof of Lemma 17. Since A, B are neighboring there exists exactly two consecutive entries for which A, B differ, denoted as (e^*, i^*) and $(e^*, i^* + 1)$. Let $\epsilon := A_{e^* i^*} - B_{e^* i^*}$, by the Definition 4 of FootRule distance, we get that $\mathrm{d_{FR}}(A, B) = |\epsilon|$. Without loss of generality we consider $\epsilon > 0$ (the case $\epsilon < 0$ symmetrically follows). We also denote with O_i the set of elements $O_i := \{e \neq e^\star$ such that $I^A_e = i\}$ and with I^A_e, I^B_e the indices in Step 6 of Algorithm 1.

Since A, B are neighboring, the e-th row of A and the e-th row of B are identical for all $e \neq e^\star$. As a result, $I^A_e = I^B_e$ for all $e \neq e^\star$. Furthermore the neighboring property implies that even for e^*, $\sum_{s=1}^{i} A_{e^\star s} = \sum_{s=1}^{i} B_{e^\star s}$ for all $i \neq i^\star$ and thus $\Pr\left[I^A_{e^\star} = i \wedge I^B_{e^\star} = j\right] = 0$ for $(i, j) \neq (i^\star, i^\star + 1)$. Now notice that

$$\begin{aligned}
\Pr\left[I^A_{e^\star} = i^\star, I^B_{e^\star} = i^\star + 1\right] &\leq \Pr\left[\log n \cdot \sum_{s=1}^{i^\star} B_{e^\star s} \leq \alpha_e \leq \log n \cdot \sum_{s=1}^{i^\star} A_{e^\star s}\right] \\
&\leq \log n \cdot (A_{e^\star i^\star} - B_{e^\star i^\star}) = \log n \cdot \epsilon
\end{aligned}$$

Notice also that in case $I^A_{e^\star} = I^B_{e^\star}$, $\mathrm{d_{KT}}(\pi_A, \pi_B) = 0$. This is due to the fact that in such a case $I^A_e = I^B_e$ for all $e \in U$ and the fact that ties are broken lexicographically. As a result,

$$\begin{aligned}
\mathbb{E}\left[\mathrm{d_{KT}}(\pi_A, \pi_B)\right] &= \Pr\left[I^A_{e^\star} \neq I^B_{e^\star}\right] \cdot \mathbb{E}\left[\mathrm{d_{KT}}(\pi_A, \pi_B) |\ I^A_{e^\star} \neq I^B_{e^\star}\right] \\
&= \Pr[I^A_{e^\star} = i^\star, I^B_{e^\star} = i^\star + 1] \cdot \mathbb{E}\left[\mathrm{d_{KT}}(\pi_A, \pi_B) |\ I^A_{e^\star} = i^\star, I^B_{e^\star} = i^\star + 1\right] \\
&\leq \epsilon \log n \cdot \left(\mathbb{E}\left[|O_{i^\star}|\right] + \mathbb{E}\left[|O_{i^\star+1}|\right]\right)
\end{aligned}$$

where the last inequality follows by the fact that once $I^A_{e^*} = i^*$ and $I^B_{e^*} = i^* + 1$, the element e^* can move at most by $|O_{i^*}| + |O_{i^*+1}|$ positions and the fact that $I^A_{e^*}, I^B_{e^*}$ and $|O_{i^*}|, |O_{i^*+1}|$

are independent random variables.

We complete the proof we providing a bound on $\mathbb{E}\left[|O_i|\right]$. Notice that for $e \in U/\{e^*\}$,

$$\Pr[e \in O_i] \leq \Pr\left[\log n \sum_{s=1}^{i-1} A_{es} \leq \alpha_e \leq \log n \sum_{s=1}^{i} A_{es}\right] \leq \log n \cdot A_{ei}$$

which implies that $\mathbb{E}\left[|O_i|\right] \leq \log n \sum_{e \neq e^\star} A_{ei} \leq 2 \log n$. Finally we overall get,

$$\mathbb{E}\left[\mathrm{d_{KT}}(\pi_A, \pi_B)\right] \leq 4 \log^2 n \cdot \mathrm{d_{FR}}(A, B)$$

◀

Proof of Lemma 12. Given the doubly stochastic matrices A, B, let the sequence $A = A^0, A^1, \dots, A^T = B$ of neighboring stochastic matrices ensured by Claim 15. Now let $\pi^0, \pi^1, \dots, \pi^T$ the sequence of permutations that the randomized rounding of Algorithm 1 produces given as input the sequence $A = A^0, A^1, \dots, A^T = B$. Notice that,

$$\mathbb{E}\left[\mathrm{d_{KT}}(\pi^A, \pi^B)\right] \leq \sum_{t=1}^{t} \mathbb{E}\left[\mathrm{d_{KT}}(\pi^t, \pi^{t-1})\right] \leq 4 \log^2 n \cdot \sum_{t=1}^{T} \mathrm{d_{FR}}(A^t, A^{t-1}) = 4 \log^2 n \cdot \mathrm{d_{FR}}(A, B)$$

where the first inequality follows by the triangle inequality, the second by Lemma 17 and the last equality by Case 4 of Claim 15. ◀

C Omitted Proofs of Section 5

C.1 Omitted Proofs of Section 5.1

Proof of Claim 23. Let $X_{ee'}^{AB} = 1$ if (e, e') is inverted pair for the matrices A, B and 0 otherwise (respectively for $X_{ee'}^{AC}, X_{ee'}^{BC}$). By a short case study one can show that once $X_{ee'}^{AB} = 1$ then $X_{ee'}^{AC} + X_{ee'}^{BC} \geq 1$ which directly implies Claim 23. ◀

Proof of Lemma 24. We construct a doubly stochastic matrix A' for which the following properties hold,

1. The entries of A' are multiples of $\frac{1}{r}$.
2. $\mathrm{d_{FR}}(A, B) = \mathrm{d_{FR}}(A, A') + \mathrm{d_{FR}}(A', B)$.
3. $\mathrm{d_{KT}}(A, A') \leq 2r^2 \cdot \mathrm{d_{FR}}(A, A')$.

Once the above properties are established, Lemma 24 follows by repeating the same construction until matrix B is reached and by using the fact that the *fractional Kendall-Tau distance* of Definition 22 satisfies the triangle inequality.

Before proceeding with the construction of A', we present the following corollary that follows by an easy exchange argument.

▶ **Corollary 26.** *Let the stochastic matrices A, B with entries multiples of $1/r$, the values f_{ij}^e of the optimal solution in the linear program of Definition 4 (the min-cost transportation problem defining the FootRule distance $\mathrm{d_{FR}}(A, B)$) are multiples of $1/r$.*

In order to construct the matrix A' satisfying the Properties 1-3, we consider three different classes of the entries (e, i). In particular, we call an entry (e, i).

1. *right* if and only if $f_{ij}^e > 0$ for some $j > i$.

2. *left* if and only if $f^e_{ij} > 0$ for some $j < i$.
3. *neutral* if and only if $f^e_{ij} = 0$ for all $j \neq i$.

Note that the above classes do not form a partition of the entries since an entry (e, i) can be both *left* and *right* at the same time.

▶ **Corollary 27.** *Given two doubly stochastic matrices $A \neq B$, there exist entries (e, i) and (e', j) such that*
1. $j > i$
2. *the entry (e, i) is right*
3. *the entry (e', j) is left*
4. *the entry (α, ℓ) is neutral for all $\alpha \in U$ and $\ell \in \{i + 1, j - 1\}$*

We construct the matrix A' from matrix A as follows. Consider two entries (e, i) and (e', j) with the properties that Corollary 27 illustrates. The doubly stochastic matrix A' is constructed by moving $1/r$ mass from entry (e, i) to entry (e, j) and by moving $1/r$ mass from entry (e', j) to entry (e', i). More formally,

$$A'_{\alpha\ell} = \begin{cases} A_{\alpha\ell} - \frac{1}{r} & \text{if } (\alpha, \ell) = (e, i) \\ A_{\alpha\ell} - \frac{1}{r} & \text{if } (\alpha, \ell) = (e', j) \\ A_{\alpha k} + \frac{1}{r} & \text{if } (\alpha, \ell) = (e', i) \\ A_{\alpha\ell} + \frac{1}{r} & \text{if } (\alpha, \ell) = (e, j) \\ A_{\alpha\ell} & \text{otherwise} \end{cases}$$

Up next we establish the fact that $\mathrm{d_{FR}}(A, B) = \mathrm{d_{FR}}(A, A') + \mathrm{d_{FR}}(A', B)$.

▷ Claim 28. $\mathrm{d_{FR}}(A', A) = 2|j - i|/r$ and $\mathrm{d_{FR}}(A', B) = \mathrm{d_{FR}}(A, B) - 2|j - i|/r$.

Proof. The fact that $\mathrm{d_{FR}}(A', A) = 2|j - i|/r$ is trivial. We thus focus on showing that $\mathrm{d_{FR}}(A', B) = \mathrm{d_{FR}}(A, B) - 2|j - i|/r$.

Since (e, i) is *right*, there exists an index $\ell(i) > i$ such that $f^e_{i\ell(i)} > 0$. Moreover $f^e_{i\ell(i)} \geq 1/r$ since $f^e_{i\ell(i)}$ is multiple of $1/r$. Notice that $\ell(i) \neq \ell$ for $\ell \in \{i + 1, j - 1\}$ since all the entries (α, ℓ) are *neutral* (otherwise $\sum_{\alpha \in U} B_{\alpha\ell} > 1$). As a result, transfering $1/r$ mass from entry (e, i) to entry (e, j) decreases the FootRule distance between A and B by $1/r \cdot |i - j|$ since the *final destination* of the $1/r$ mass is the entry $(e, \ell(i))$ that is on the right of entry (e, j), $\ell(i) \geq j$. The claim follows by applying the exact same argument for (e', j). ◀

We now establish the last property that is $\mathrm{d_{KT}}(A, A') \leq 2r^2 \cdot \mathrm{d_{FR}}(A, A')$.

▷ Claim 29. $\mathrm{d_{KT}}(A', B) \leq 4r \cdot |i - j|$

Proof. Notice that apart from e, e', the r-index of each element is the same in both A and A' ($I^A_\alpha = I^{A'}_\alpha$ for all $\alpha \in U \setminus \{e, e'\}$). As a result, by Definition 22, we get that the only inverted pairs can be of the form (e, α) or (e', α).

In case $I^A_e \leq i - 1$ then $I^A_e = I^{A'}_e$ and there is no inverted pair of the form (e, α). In case $I^A_e = i$ then $i \leq I^{A'}_e \leq j$ and any element α with $I^A_\alpha = I^{A'}_\alpha \in \{1, i-1\} \cup \{j+1, n\}$ cannot form an inverted pair with e. As a result, a pair (e, α) can be inverted only if $i \leq I^A_\alpha = I^{A'}_\alpha \leq j$. Since the entries of A are multiples of $1/r$ and A is doubly stochastic, there are at most r positive entries at each column of A. As a result, there are at most $r \cdot (j - i + 1)$ inverted pairs of the form (e, α). With the symmetric argument one can show that there are at most

$r \cdot |j-i+1|$ of the form (e', α). Overall there are at most $2r \cdot |j-i+1|$ inverted pairs between A and A' that are less than $4r \cdot |j-i|$ since $j > i$. ◀

◀

Proof of Lemma 25. Since $B^t_{e_t} \geq 1/r$, the r-index of element e_t in matrix B^t is 1, $I^{B^t}_{e_t} = 1$. We first show that,

$$\mathrm{d}_{\mathrm{KT}}\left(\pi^t, \pi^{t-1}\right) + \mathrm{d}_{\mathrm{KT}}\left(\pi^t, B^t\right) - \mathrm{d}_{\mathrm{KT}}\left(\pi^{t-1}, B^t\right) \leq r$$

To simplify notation let k_t the position of e_t in π^{t-1}. Notice that $\mathrm{d}_{\mathrm{KT}}\left(\pi^t, \pi^{t-1}\right) = k_t - 1$. Out of the $k_t - 1$ elements lying on the left of e_t in π^{t-1} there are most $r-1$ elements α with $I^{B^t}_{\alpha} = 1$ (these elements must admit $B^t_{\alpha 1} \geq 1/r$). The rest of the $k_t - 1$ elements admit r-index $I^{B^t}_{\alpha} \geq 2$ and thus form inverted pairs with e_t when considering π^{t-1} and B^t. When e_t moves to the first positions (permutation π^t) these inverted pairs are deactivated ($I^{B^t}_{e_t} = 1$) and new inverted pairs are created between e_t and α with $I^{B^t}_{\alpha} = 1$, but these new inverted pairs are at most r (for any element α with $I^{B^t}_{\alpha}$, $B^t_{\alpha} \geq 1/r$). Also notice no additional inverted pairs (e, α) (with $e \neq e_t$) are created since the order between all the other elements is the same in π^t and π^{t-1}. Overall,

$$\underbrace{\mathrm{d}_{\mathrm{KT}}\left(\pi^t, \pi^{t-1}\right)}_{k_t - 1} + \underbrace{\mathrm{d}_{\mathrm{KT}}\left(\pi^t, B^t\right) - \mathrm{d}_{\mathrm{KT}}\left(\pi^{t-1}, B^t\right)}_{\leq -k_t + 1 + r} \leq r$$

Combining the above inequality with $\mathrm{d}_{\mathrm{KT}}\left(\pi^{t-1}, B^t\right) - \mathrm{d}_{\mathrm{KT}}\left(\pi^{t-1}, B^{t-1}\right) \leq \mathrm{d}_{\mathrm{KT}}\left(B^t, B^{t-1}\right)$ which follows from the triangle inequality we get,

$$\mathrm{d}_{\mathrm{KT}}\left(\pi^t, \pi^{t-1}\right) + \mathrm{d}_{\mathrm{KT}}\left(\pi^t, B^t\right) - \mathrm{d}_{\mathrm{KT}}\left(\pi^{t-1}, B^{t-1}\right) \leq \mathrm{d}_{\mathrm{KT}}\left(A^t, B^{t-1}\right) + r.$$

Finally a telescopic sum gives $\sum_{t=1}^T \mathrm{d}_{\mathrm{KT}}\left(\pi^t, \pi^{t-1}\right) \leq \sum_{t=1}^T \mathrm{d}_{\mathrm{KT}}\left(B^t, B^{t-1}\right) + r \cdot T + \mathrm{d}_{\mathrm{KT}}(\pi^0, B^0) - \mathrm{d}_{\mathrm{KT}}(\pi^T, B^T)$ where $\mathrm{d}_{\mathrm{KT}}(\pi^0, B^0) = 0$. ◀

C.2 Proof of Lemma 19

We prove the existence of an optimal solution $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ for the linear program of Definition 18 for which the entries of each matrix $\hat{A}^t$ are multiples of $1/r$ though the design of an optimal greedy algorithm illustrated in Algorithm 3.

The fact that Algorithm 3 produces a solution with entries that multiples of $1/r$ easily follows. Algorithm 3 starts with an integral doubly stochastic matrices ($\hat{A}^0 = \pi^0$) and always moves $1/r$ mass from entry to entry. The optimality of Algorithm 3 is established in Lemma 30 the proof of which is presented in the next section since it is quite technically complicated. However the basic idea of the algorithms is very intuitive, once $\hat{A}^{t-1}_{e_t} = 0$ Algorithm 3 moves $1/r$ mass of e_t from its leftmost position (with mass greaer than $1/r$), denoted as Pos of Step 5. At this point of time, Algorithm 3 has violated the column-stochasticity constraints, $1 + 1/r$ for the first column and $1 - 1/r$ for the Pos-th column and Algorithm 3 must move at total of $1/r$ mass from the first position to next positions until $1/r$ mass reaches the Pos position and column-stochasticity is restored (Step 8). Once Algorithm 3 detects an element with aggregated mass (until position j) $\geq 2/r$, it can safely move $1/r$ of each mass to position $j+1$ since even if this element appears at some point in the future only $1/r$ is necessary to satisfy the constraint $A^t_{e_t 1} \geq 1/r$ and thus the rest is redundant (Step 11). In case such an element does not exist, Algorithm 3 moves the (useful) $1/r$ mass of the element appearing the furthest in the remaining sequence $\{e_t, \ldots, e_T\}$, which is exactly the same optimal *eviction policy* that the well-studied $k-$Paging suggests.

Algorithm 3 An Optimal Greedy Algorithm for the LP of Definition 18

Input: The initial permutation π^0 and the sequence of elements $e_1, \dots, e_T \in U$
Output: An optimal solution of a linear program of Definition 18 where the entries of $\hat{A}^t$ are multiples of $1/r$.

Initially $\hat{A}^0 \leftarrow \pi_0$
for all rounds $t = 1$ to T **do**
  $\hat{A}^t \leftarrow \hat{A}^{t-1}$
  if $\hat{A}^t_{e_t 1} < 1/r$ **then**
    //Move $1/r$ mass of e_t to the first position
      Pos $\leftarrow \mathrm{argmin}_{1 \le i \le n} \{A^t_{ei} \ge 1/r\}$
      $\hat{A}^t_{e1} \leftarrow \hat{A}^t_{e1} + 1/r, \hat{A}^t_{e\mathrm{Pos}} \leftarrow \hat{A}^t_{e\mathrm{Pos}} - 1/r$
    //Restore the column-stochasticity constraints from left to right
    for $j = 1$ to $\mathrm{Pos} - 1$ **do**
      if there exists $e \in U$ with $\sum_{s=1}^{j} \hat{A}^t_{es} \ge 2/r$ and $\hat{A}^t_{es} \ge 1/r$ **then**
        //Move $1/r$ of its (redundant) mass to the next position
          $\hat{A}^t_{ej} \leftarrow \hat{A}^t_{ej} - 1/r$, $\hat{A}^t_{ej} \leftarrow \hat{A}^t_{ej} + 1/r$
      else
        //Move the $1/r$ mass, of the element appearing furthest in the future, to the next position
            $e^\star \in U \leftarrow$ the element with $\hat{A}^t_{e^\star j} = 1/r$ furthest in $\{e_{t+1}, \dots, e_T\}$
            $\hat{A}^t_{e^\star j} \leftarrow \hat{A}^t_{e^\star j} - 1/r$, $\hat{A}^t_{e^\star j} \leftarrow \hat{A}^t_{e^\star j} + 1/r$
      end if
    end for
  end if
end for
return $\hat{A}_1, \dots, \hat{A}_T$

▶ **Lemma 30.** *Algorithm 3 produces an optimal solution $\hat{A}^0 = \pi^0, \hat{A}^1, \dots, \hat{A}^T$ for the linear program of Definition 18 while the entries of each $\hat{A}^t$ are multiples of $1/r$.*

C.2.1 Proof of Lemma 30

At first notice that the linear program of Definition 4 defining the FootRule distance $\mathrm{d_{FR}}(A, B)$ between the stochastic matrices A, B can take the following equivalent form.

$$
\begin{aligned}
\min \quad & \sum_{e \in U} \sum_{i=1}^{n} (P_{ei} + M_{ei}) \\
\text{s.t} \quad & P_{ei} - M_{ei} = \sum_{s=1}^{i} (A_{es} - B_{es}) \qquad \text{for all } e \in U \text{ and } i = 1, \dots, n \\
& P_{ei}, M_{ei} \ge 0 \qquad\qquad\qquad\qquad\quad \text{for all } e \in U \text{ and } i = 1, \dots, n
\end{aligned}
$$

This is due to the fact that $(P_{ei} + M_{ei})$ takes the value $|\sum_{s=1}^{i} (A_{es} - B_{es})|$ and it is not hard to prove that $\sum_{e \in U} |\sum_{i=1}^{n} (A_{es} - B_{es})|$ equals $\mathrm{d_{FR}}(A, B)$. As a result, the linear program of Definition 18 takes the following equivalent form,

$$
\begin{aligned}
\min \quad & \sum_{t=1}^{T}\sum_{e=1}^{n}\sum_{i=1}^{n}\left(P_{ei}^t + M_{ei}^t\right) & \\
\text{s.t} \quad & P_{ei}^t - M_{ei}^t = \sum_{s=1}^{i}\left(\hat{A}_{es}^t - \hat{A}_{es}^{t-1}\right) & \text{for all } e \in U \text{ and } t \in \{1,T\} \\
& \sum_{e\in U}\hat{A}_{ei}^t = 1 & \text{for all } t \in \{1,T\} \text{ and } i \in \{1,n\} \\
& \sum_{i=1}^{n}\hat{A}_{ei}^t = 1 & \text{for all } t \in \{1,T\} \text{ and } e \in U \\
& \hat{A}_{e_t 1}^t \geq 1/r & \text{for all } t \in \{1,T\} \\
& \hat{A}_0 = \pi_0 & \\
& \hat{A}_{ei}^t, P_{ei}^t, M_{ei}^t \geq 0 &
\end{aligned}
$$

In Definition 31 we construct n different linear programs admitting the property that the sum of their optimal values acts as a lower bound on the optimal solution of the linear program of Definition 18 and will help us establish the optimality of Algorithm 3.

▶ **Definition 31.** *For each $1 \leq i \leq n$ consider the following linear program,*

$$
\begin{aligned}
\mathit{min} \quad & \sum_{t=1}^{T}\sum_{e=1}^{n}\left(X_{ei}^t + Y_{ei}^t\right) & \\
\mathit{s.t} \quad & X_{ei}^t - Y_{ei}^t = B_{ei}^t - B_{ei}^{t-1} & \textit{for all } e \in U \textit{ and } t \in \{1,T\} \\
& \sum_{e\in U} B_{ei}^t = i & \textit{for all } t \in \{1,T\} \\
& B_{e_t i}^t \geq 1/r & \textit{for all } t \in \{1,T\} \\
& B_{ei}^0 = \sum_{s=1}^{i}\hat{A}_{es}^0 & \textit{for all } e \in U \\
& X_{ei}^t, Y_{ei}^t, B_{ei}^t \geq 0 & \textit{for all } e \in U \textit{ and } t \in \{1,T\}
\end{aligned}
$$

Let $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ denote the optimal solution of the linear program of Definition 18. Notice that,

$$\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(\hat{A}^t, \hat{A}^{t-1}) \geq \sum_{i=1}^{n}\sum_{t=1}^{T}\sum_{e\in U}\left(X_{ei}^t + Y_{ei}^t\right)$$

where X_{ei}^t, Y_{ei}^t denote the values of the respective variables in the optimal solution of the i-th linear program in Definition 31. This is due to the fact that setting $B_{ei}^t = \sum_{s=1}^{i} A_{es}^t$ produces a feasible solution for the i-th linear program in Definition 31. We will prove that for the sequence $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ produced by Algorithm 3,

$$\sum_{t=1}^{T} \mathrm{d}_{\mathrm{FR}}(\hat{A}^t, \hat{A}^{t-1}) = \sum_{i=1}^{n}\sum_{t=1}^{T}\sum_{e\in U}\left(X_{ei}^t + Y_{ei}^t\right)$$

which implies optimality.

To this end, we present a very natural interpretation of the i-th linear program in Definition 31 that will help us design an optimal greedy algorithm (Algorithm 4) for solving the linear programs of Definition 31. Each of the linear programs of Definition 31 can be viewed as a *fractional version* of the well-known $k-$Paging []. In Definition 32 we provide this alternative and more intuitive definition for the linear programs of Definition 31.

▶ **Definition 32** (Fractional Paging). *Given a sequence of elements $e_1, \dots, e_T$ and an initial vector S^0 such that $|S^0| = n$, $0 \le S^0_e \le 1$ and $\sum_{e \in U} S^0_e = i$. Compute a sequence of vectors $S_1, \dots, S_T$ such that*

1. $0 \le S^t_e \le 1$
2. $\sum_{e \in U} S^t_e = i$
3. $S^t_{e_t} \ge 1/r$ *for* $1 \le t \le T$

and the quantity $\sum_{t=1}^{T} \sum_{e \in U} |S^t_e - S^{t-1}_e|$ *is minimized.*

In Algorithm 4 we present a generalization the classical greedy eviction policy (*evict the page arriving latter in the future*) that is the optimal policy for the original paging problem.

Algorithm 4 Greedy Algorithm for Fractional Paging

Input: An initial vector S_0 and a sequence of elements $e_1, \dots, e_T \in U$
Output: A sequence of vectors $S_1, \dots, S_T$.

```
for t = 1 to T do
  S^t ← S^{t-1}
  S^t_{e_t} ← S^t_{e_t} + min(1/r − S^t_{e_t})
  for each element e ∈ U do
    //Remove first the redundant mass.
    if S^t_e ≥ 2/r and ∑_{e∈U} S^t_e ≥ i then
      S^t_e ← S^t_e − min(S^t_{e_t} − 1/r, ∑_{e∈U} S^t_e − i)
    end if
    //If redundant mass is not enough, remove mass from the elements arriving latter
    in the future.
    t*(e) ← the first time s ≥ t such that e_s = e.
    Sort the elements in decreasing order according to t*(e).
    for all elements e ∈ U (according to the previous ordering) do
      S^t_e ← S^t_e − min(S^t_e, ∑_{e∈U} S^t_e − i).
    end for
  end for
end for
return S_1, ..., S_T
```

▶ **Lemma 33.** *Algorithm 4 is optimal for the fractional paging.*

Proof. Let O^t denote the vector of the optimal solution of the Fractional Paging of Definition 32 at round t, for $1 \le t \le T$. Without loss of generality we assume that $O^t_\alpha \ge O^{t-1}_\alpha$ for $\alpha = e_t$ and $O^t_\alpha \le O^{t-1}_\alpha$ for $\alpha \ne e_t$.

We first construct a sequence of vectors S^t for $1 \le t \le T$ such that the vector S^1 agrees with Algorithm 4 and the cost of the vector sequence $S^1, \dots, S^T$ is optimal. Once this established the proof follows inductively.

Before presenting the construction we partition the elements into the following 3 classes. The set *neutral* denoted by N_t denotes the elements e for which $S^t_e = \min(1/r, O^t_e)$. The set of *greater elements at round* t, denoted by G_t, which are all elements $e \notin N_t$ and $S^t_e \ge O^t_e$. Finally we have the set of *smaller elements* L_t which are all $e \notin N_t$ and $S^t_e < O^t_e$. The vector sequence $\{S^t\}_{2 \le t \le T}$ is inductively defined as follows: **For each round $t \ge 2$,**

1. If $O^t_e \ge O^{t-1}_e$

- If $e \in N_{t-1}$ then $S_e^t = \min(1/r, O_e^t)$.
- If $e \in G_{t-1}$ then $S_e^t = S_e^{t-1} + \min\big(\min(O_e^t, 1/r) - S_e^{t-1}, 0\big)$.
- If $e \in L_{t-1}$ then $S_e^t = S_e^{t-1} + \min(O_e^t - O_e^{t-1}, 1/r - S_e^{t-1})$.

2. If $O_e^t < O_e^{t-1}$
- If $e \in N^{t-1}$ then $S_e^t = \min(1/r, O_e^t)$.
- If $e \in G^{t-1}$ then $S_e^t = S_e^{t-1} - O_e^{t-1} + O_e^t$.
- If $e \in L^{t-1}$ then $S_e^t = S_e^{t-1} - \max(S_e^{t-1} - O_e^t, 0)$.

Finally in case $\sum_{e \in U} S_e^t > i$, we additionally subtract total amount of $\sum_{e \in U} S_e^t - i$ from the elements $S_e^t \geq O_e^t$. As a result, the cost of round t is

$$\sum_{e \in S_e^t \geq S_e^{t-1}} (S_e^t - S_e^{t-1}) + \sum_{e \in S_e^t \leq S_e^{t-1}} (S_e^{t-1} - S_e^t) + \sum_{e \in U} S_e^t - i = 2 \sum_{e \in S_e^t \geq S_e^{t-1}} (S_e^t - S_e^{t-1})$$

By the definition of S^t,

$$2 \sum_{e \in S_e^t \geq S_e^{t-1}} (S_e^t - S_e^{t-1}) = 2 \sum_{e \in O_e^t \geq O_e^{t-1}} (S_e^t - S_e^{t-1}) \leq 2 \sum_{e \in O_e^t \geq O_e^{t-1}} (O_e^t - O_e^{t-1}) = ||O^t - O^{t-1}||_1$$

As a result, we overall get that $\sum_{t=1}^T ||S^t - S^{t-1}||_1 \leq \sum_{t=1}^T ||O^t - O^{t-1}||_1$.

Up next we prove that the solution $S_1, \ldots, S_T$ is a feasible solution for *fractional paging*.

At first observe that an element e can only go from the state form the state of *greater* to the state of *neutral* and from the state of *smaller* to the state of *neutral*. Moreover observe that *once an element becomes neutral it remains neutral forever.*

The only case that the constructed solution $S^1, \ldots, S^T$ is not feasible is by having $e \in L^t$ with $e = e_t$ for some round t (otherwise $S_e^t \geq O_e^t \geq 1/r$). Combining the latter with the previous observation we get that $e \in L^\ell$ for all $1 \leq \ell \leq t$. Moreover $S_e^t < 1/r$.

We consider the mutually exclusive cases $S_e^1 < 1/r$ and $S_e^1 \geq 1/r$.

Let us start with $S_e^1 < 1/r$. By Algorithm 4, we get that $e \neq e_1$ and thus $O_e^1 \leq S_e^0$ ($S^0 = O^0$) and since $e \in L^1$, we get that $S_e^1 < S_e^0$. Since $S_e^1 < 1/r$ and $S_e^1 < S_e^0$ by Steps $4 - 8$ of Algorithm 4, we get that for $\alpha \in U$, $S_\alpha^1 \leq 1/r$ (all the redundant mass is removed before useful is removed). Moreover by Steps $11 - 14$ we get that for all $\alpha \in G^1$, $t^*(\alpha) < t^*(e)$[3]. Finally notice that in case $S_\alpha^{t-1} < 1/r, \alpha = e_t$ and $\alpha \in G^{t-1}$ then α becomes neutral, $\alpha \in N^t$. This implies that until round $t^*(e)$ all elements $\alpha \in G^1$ have become neutral. As a result, at round $t^\star(e)$ all elements $\alpha \in G^1$ have become neutral which means that for all elements $\alpha \in U/\{e\}$, $S_\alpha^t \leq O_\alpha^t$. If $e \in L^t$ then $\sum_{\alpha \in U} S_\alpha^t < \sum_{\alpha \in U} O_\alpha^t$ which is a contradiction. Thus $S_e^{t^*(e)} \geq 1/r$.

In either the case $S_e^1 \geq 1/r$ or $S_e^{t^*(e)} \geq 1/r$ (that follows in case $S_e^1 < 1/r$) we get that there exists an $\ell \leq t$ such that $S_e^{\ell-1} \geq 1/r$ and $S_e^\ell < 1/r$. Since $e \in L^{\ell-1}$, we get that $S_e^\ell = S_e^{\ell-1} - \max(S_e^{\ell-1} - O_e^\ell, 0)$ which implies that $S_e^\ell = O_e^\ell$ and thus e becomes neutral. ◀

[3] Let $\alpha \in G_1$ with $t^*(\alpha) > t^*(e)$ then by Step 11 and 12 of Algorithm 4 $S_\alpha^1 = 0$, something that cannot be true, since $\alpha \in G_1$.

We conclude the section with the proof of Lemma 30. We set $B^t_{ei} = \sum_{s=1}^{i} \hat{A}^t_{es}$ where $\hat{A}^0 = \pi^0, \hat{A}^1, \ldots, \hat{A}^T$ are the stochastic matrices produced by Algorithm 3. We show that for $i = 1, \ldots, n$ the constructed sequence $B^0_{ei}, \ldots, B^T_{ei}$ is *consistent* with Algorithm 4, meaning that the $B^0_{ei}, \ldots, B^T_{ei}$ could have been the output of Algorithm 4. As a result,

1. Each $\sum_{t=1}^{T} \sum_{e \in U} (X^t_{ei} + Y^t_{ei})$ equals the optimal value of the i-th linear program in Definition 31.
2. $\sum_{i=1}^{n} \sum_{t=1}^{T} \sum_{e \in U} (X^t_{ei} + Y^t_{ei}) = \sum_{t=1}^{T} \mathrm{d_{FR}}(\hat{A}^t, \hat{A}^{t-1})$ since we can set $P^t_{ei} = X^t_{ei}$ and $M^t_{ei} = Y^t_{ei}$ ($B^t_{ei} = \sum_{s=1}^{i} \hat{A}^t_{es}$).

The latter two properties establish the optimality of Algorithm 3.

We now prove that the sequence $B^0_{ei}, \ldots, B^T_{ei}$ where $B^t_{ei} = \sum_{s=1}^{i} \hat{A}^t_{es}$ is consistent with Algorithm 4. We emphasize that we prove this under the condition that the initial vector B^0_{ei} is a $0-1$ vector which simplifies a lot the actions of Algorithm 4. In particular and an easy induction argument reveals that when the initial vector B^0_{ei} is a $0-1$ vector then the vectors produced by Algorithm 4 admit entries with multiples of $1/r$, something that is obviously true for the sequence $B^0_{ei}, \ldots, B^T_{ei}$ since $B^t_{ei} = \sum_{s=1}^{i} \hat{A}^t_{es}$.

Let $B^t_{e_t i} = B^{t-1}_{e_t i}$, this is the case where $B^{t-1}_{e_t i} \geq 1/r$. In order to ensure consistency with Algorithm 4, we need to show that $B^t_{ei} = B^{t-1}_{ei}$ for all $e \in U$ (see Step 3 of Algorithm 4). By the fact that $B^t_{e_t i} = B^{t-1}_{e_t i}$ we get that $\sum_{s=1}^{i} \hat{A}^t_{e_t s} = \sum_{s=1}^{i} \hat{A}^{t-1}_{e_t s}$ which by Algorithm 3 implies that $\hat{A}^{t-1}_{ej} \geq 1/r$ for some $j \leq i$. The latter together with Algorithm 3 (see Step 9 of Algorithm 3) implies that $\sum_{s=1}^{i} \hat{A}^t_{ei} = \sum_{s=1}^{i} \hat{A}^{t-1}_{ei}$ for all $e \in U$. Thus, $B^t_{ei} = B^{t-1}_{ei}$.

Let $B^t_{e_t i} = B^{t-1}_{e_t i} + 1/r$, this is the case where $B^{t-1}_{e_t i} = 0$. To establish consistency with Algorithm 4 we need to prove that there exists a unique e^* with $B^t_{e^* i} = B^{t-1}_{e^* i} - 1/r$ and one of the following holds,

1. $B^{t-1}_{e^* i} \geq 2/r$ (Condition 6 in Algorithm 4)
2. $B^{t-1}_{ei} \leq 1/r$ for all $e \in U$ and $B^{t-1}_{e^* i} = 1/r$ is the element appearing furthest in the sequence $\{e_{t+1}, \ldots, e_T\}$ (Condition 6 in not met and Algorithm 4 continues in Steps $11-14$).

Step $9-14$ of Algorithm 3 guarantees that the existence of a unique element e^* such that $\sum_{s=1}^{i} \hat{A}^t_{e^* s} = \sum_{s=1}^{i} \hat{A}^{t-1}_{e^* s} - 1/r$ which implies the existence of a unique element e^* with $B^t_{e^* i} = B^{t-1}_{e^* i} - 1/r$ since $B^t_{e^* i} = \sum_{s=1}^{i} \hat{A}^t_{e^* s}$. In case $\sum_{s=1}^{i} \hat{A}^{t-1}_{e^* s} \geq 2/r$ then we are done. So let us assume that $\sum_{s=1}^{i} \hat{A}^{t-1}_{e^* s} = 1/r$. This implies that $\sum_{s=1}^{i} \hat{A}^{t-1}_{es} \leq 1/r$ for all $e \in U$ since otherwise Algorithm 3 would have moved an element e' with $\sum_{s=1}^{i} \hat{A}^{t-1}_{e' s} \geq 2/r$ (see Step 11 of Algorithm 3). The fact that e^* is the element with $\sum_{s=1}^{i} \hat{A}^{t-1}_{e^* s} = 1/r$ appearing furthest in the sequence $\{e_{t+1}, \ldots, e_T\}$ is ensured by Step 14 of Algorithm 3.