Working Paper No. 20-29

Batching and Optimal Multi-stage Bipartite Allocations

Yiding Feng
Northwestern University

Rad Niazadeh
University of Chicago Booth School of Business

Batching and Optimal Multi-stage Bipartite Allocations

Yiding Feng
Microsoft Research New England, Cambridge, MA, yidingfeng@microsoft.com
Rad Niazadeh
University of Chicago Booth School of Business, Chicago, IL, rad.niazadeh@chicagobooth.edu

In several applications of real-time matching of demand to supply in online marketplaces, the platform allows for some latency to batch the demand and improve the efficiency of the resulting matching. Motivated by these applications, we study the optimal trade-off between batching and inefficiency in the context of designing robust online allocations. As our base model, we consider K-stage variants of the classic vertex weighted bipartite b-matching in the adversarial setting, where online vertices arrive stage-wise and in K batches — in contrast to online arrival. Our main result for this problem is an optimal $\left(1-(1-1/K)^K\right)$-competitive (fractional) matching algorithm, improving the classic $(1-1/e)$ competitive ratio bound known for its online variant (Mehta et al., 2007; Aggarwal et al., 2011). We also extend this result to the general problem of multi-stage configuration allocation with free-disposals (Devanur et al., 2016), which is motivated by the display advertising application in the context of video streaming platforms.

Our main technique at high-level is developing algorithmic tools to vary the trade-off between "greedyness" and "hedging" of the matching algorithm across stages. We rely on a particular family of convex-programming based matchings that distribute the demand in a specifically balanced way among supply in different stages, while carefully modifying the balancedness of the resulting matching across stages. More precisely, we identify a sequence of *polynomials with decreasing degrees* to be used as strictly concave regularizers of the maximum weight matching linear program to form these convex programs. At each stage, our fractional multi-stage algorithm returns the corresponding regularized optimal solution as the matching of this stage (by solving the convex program). By providing structural decomposition of the underlying graph using the optimal solutions of these convex programs and recursively connecting the regularizers together, we develop a new multi-stage primal-dual framework to analyze the competitive ratio of this algorithm. We further show this algorithm is optimal competitive, even in the unweighted case, by providing an upper-bound instance in which no online algorithm obtains a competitive ratio better than $\left(1-(1-1/K)^K\right)$. For the extension to multi-stage configuration allocation, we introduce a novel extension of our regularized convex program that provides separate regularization at different "price levels". Despite the lack of a relevant graph decomposition in this extension, in contrast to our base model, we show how we can directly use convex duality to set up a primal-dual analysis framework for our new algorithm.

1. Introduction

With the pervasiveness of online platforms and web-based services, internet advertising has grown to become a major component of today's economy. Worldwide spending of internet advertising stood at an estimated 521 billion U.S. dollars in 2021 — a figure that is forecast to constantly increase in the coming years, reaching a total of 876.1 billion U.S. dollars by 2026.[1] The combination of the economic impact and the inherent flexibility of internet advertising has helped with its prevalence as a modern market. At the same time, it has also given rise to numerous revenue management challenges and paradigms, some of which pertain to real-time algorithmic allocation of inventories of ad opportunities to advertisers in various search and display internet advertising scenarios, e.g., see Mehta et al. (2007); Feldman et al. (2009a); Balseiro et al. (2014); Devanur et al. (2016).

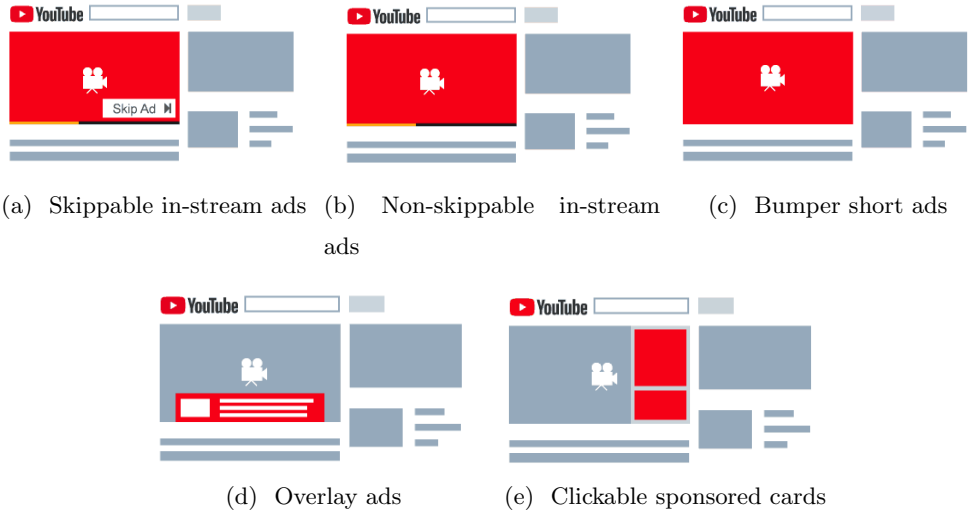
As digital consumption behaviors change, and new technologies provide new ways for users to interact with social media platforms and publishers, traditional forms of real-time internet ad allocations also evolve. A prominent example of this sort, which is one of the main motivating application behind our paper, is *in-stream video advertising*. Nowadays, in-stream video advertising is rapidly growing [2] and is playing an essential role in the business model of online video sharing platforms such as YouTube and Vimeo, on-demand/live streaming platforms such as Netflix, Roku, and Twitch, and even social video streaming platforms such as TikTok. In this context, when a user starts watching a new video session (e.g., on YouTube), an advertising opportunity to display video-ads, and hence allocating user impressions to advertisers and charging them based on per-impression rates, is created. Video-ads refer to all possible ad formats in this context such as in-stream advertising videos of different lengths, as well as the banner-like overlay and text- or image-based ads that appear over a video. See Figure 1 for specific video-ad formats that are currently used in YouTube.[3] Given these ad format varieties, the real-time ad allocation for a sequence of arriving users is done by displaying ads of one or multiple advertisers to each arriving user (or equivalently, each video session). It is important to add that such a dynamic environment is highly non-stationary (in terms of users that start watching videos) and hence enabling the platform to run a robust and prior free ad allocation policy is a common practice.

While in-stream video advertising is a real-time application, it still admits a natural form of allowance for latency in displaying ads to the users. In particular, as several of the video-ads show up in the middle (or at the end) of users' video sessions, an ad allocator can delay the time of its decision for a

[1] See https://www.statista.com/statistics/237974/online-advertising-spending-worldwide/.

[2] According to Deloitte (2021), 80% of U.S. consumers are now paying for streaming video services, up from 73% before the COVID-19 pandemic began. Using ad spending as an indicator, video advertising will likely continue to grow for the foreseeable future. Video ad spending in the U.S. is increased from 8.92 billion U.S. dollars in 2017 to 55.34 billion U.S. dollars in 2021; it is also projected to reach as high as 78.45 billion U.S. dollars by 2023. See https://www.statista.com/statistics/256272/digital-video-advertising-spending-in-the-us/.

[3] See https://support.google.com/youtube/answer/2375464?hl=en.



(a) Skippable in-stream ads (b) Non-skippable in-stream ads (c) Bumper short ads

(d) Overlay ads (e) Clickable sponsored cards

Figure 1 Different formats of video advertising in YouTube streaming; These formats can be used at different time-stamps of a YouTube video, and differ in their lengths, designs, and how users interact with them.

session (or the current chunk of a long video session) from the beginning of the video to the first time-stamp that a video-ad is required to be displayed. This flexibility provides an immense opportunity to *batch* the arriving video sessions, where a batch consists of the most recent group of users that their ad allocation decisions can be made in the same time frame. Given these batches, the goal of the ad allocator is to make allocation decisions as each batch of users arrives — in contrast to fully online decisions as each single user arrives — in order to maximize the revenue from the advertisers. This batching technique is indeed practiced in the web-based video advertising industry (YouTube, 2021).

Besides video advertising, there are also other resource allocation applications with latency allowance, for example, matching delivery requests to dispatching centers in Amazon or matching user jobs to computing servers in Azure cloud computing platform. Even in fast-paced applications such as sponsored search, the ad decisions for the arriving search queries can still be delayed in the order of milliseconds, and hence batching is possible. In all of these applications, while dividing the entire real-time sequence of demand requests into less number of batches (of larger sizes) requires more latency, it clearly helps the platform to make improved allocation decisions in terms of its objective as it has more information about the sequence of arriving demand requests.[4]

Our Theoretical Lens Motivated by the batching-and-allocation paradigm in practical applications such as video advertising, and also to theoretically characterize the the inherent trade-off between batching and inefficiency in robust and prior-free dynamic allocations, we revisit the literature on

[4] To see numbers in practice, for example, ridesharing platforms such as Uber (Uber, 2020), Lyft (Lyft, 2016) and DiDi (Zhang et al., 2017) have reported substantial efficiency improvements due to shifting from match-as-you-go to batching-and-matching. Same results are reported in other corners of gig-economy such as cloud computing markets (AWS, 2021) and online food delivery (Joshi et al., 2021). Notably, batching not only helps with improving the efficiency, it can also help with reducing non-operational forms of cost such as waste and environmental cost (CNN, 2021).

(adversarial) online bipartite allocations, originated from the seminal work of Karp et al. (1990), where the goal is to allocate the arriving demand vertices (online side) to the available supply vertices (offline side). We study several canonical settings in this literature under the *multi-stage* arrival model, where the online vertices are partitioned into $K > 0$ fixed number of batches and arrive over K stages (one batch per each stage). In this model, once a batch arrives, every information in the problem instance related to this batch (which depends on the problem's specifications) is revealed to the multi-stage allocation algorithm. The algorithm then makes irrevocable stage-wise decisions by allocating the arriving batch of vertices to the offline side at each stage.

As our base model, we consider the *multi-stage vertex weighted b-matching* (Aggarwal et al., 2011; Devanur et al., 2013). In this simple problem, online vertices arrive in $K > 0$ batches. Then the multi-stage algorithm matches the arriving batches to the offline side while respecting the capacity constraints on the offline side. The goal is to maximize the total weight of the matching.

Motivated by the video advertising application, we also consider an extension model called *multi-stage configuration allocation* (Devanur et al., 2016), which generalizes a rich set of multi-stage resource allocation problems including vertex-weighted b-matching, budgeted allocation (also known as "AdWords" (Mehta et al., 2007)), and edge-weighted online matching with free-disposal (Feldman et al., 2009b). In this problem, we have online users (each corresponding to a new video session) that arrive in $K > 0$ batches and offline advertisers that have ads to be displayed to these users so that they receive impressions from the users. Once a batch arrives, the multi-stage allocation algorithm assigns a feasible "configuration" to each user in that batch. Here, a configuration can refer to an arrangement of video-ads of various lengths and formats for a subset of advertisers, showing up at different time-stamps of the user's video session.[5] Each assigned configuration yields a certain number of impressions to each advertiser and charges the advertiser at a certain price per impression. Depending on advertisers' long-term contracts, each advertiser is willing to pay for a certain number of impressions during the decision making horizon. Given this model, the goal is to maximize the revenue (i.e., total collected prices) in the finite decision making horizon. Particularly important in this model, the set of feasible configurations can model various complex user-level constraints.[6]

We measure the performance of our algorithms by their *competitive ratio*, which is the worst-case ratio between the expected objective value of the algorithm and the in-hindsight optimal solution — also known as the optimal offline benchmark. Considering different degrees of batching by changing the number of batches K and their sizes, one extreme regime of our problem is the fully offline setting

[5] It can also be the case that a long video session is divided into chunks, and then each chunk is assigned a separate configuration, i.e., is treated as a new video session.

[6] To name a few, our model can capture exclusion (e.g., competing advertisers can not be shown in the same video), frequency capping (e.g., similar ads should not be shown back to back in the same video), and bundling (e.g., when advertisers require that all or none of a set of related ads be shown in the same video).

when there is only a single large batch containing all the demand vertices. In this extreme regime one can achieve the performance of the optimum offline benchmark. The other extreme regime is the fully online setting when each online demand vertex arrives essentially as a batch of size one (no batching). Note that any competitive ratio lower-bound for the fully online setting establishes a lower-bound for the multi-stage setting, as any algorithm in the fully online setting can be used in the multi-stage setting by simply considering any arbitrary (synthetic) arrival order over vertices in each batch. We now ask the following research question in this paper:

In the middle-ground multi-stage regime of bipartite allocation problems with batch arrivals (in contrast to fully online and offline regimes), when we have $K > 0$ number of batches, what are the optimal competitive ratios achievable by multi-stage algorithms in these problems?

Our Contributions The main contribution of this paper is the following result.

[Main Result] *For the multi-stage vertex weighted matching (resp. multi-stage configuration allocation) problem with K batches, we propose Algorithm 1 (resp. Algorithm 2) that achieves a competitive ratio of $\Gamma(K) \triangleq 1 - \left(1 - \frac{1}{K}\right)^K$. Moreover, by proposing an upper-bound instance, we show that no K-stage algorithm, fractional or integral, can have a competitive ratio better than $\Gamma(K)$ in these problems.*

The competitive ratio function $\Gamma(K) \in (1 - 1/e, 1]$ is monotone decreasing in K. It is equal to 1 at $K = 1$ and converges to $1 - 1/e$ as K goes to infinity.[7]. For the special case of two-stage vertex weighted matching ($K = 2$), our algorithm is 3/4-competitive, which matches the result of Feng et al. (2020b). Also, for three stages ($K = 3$), it has a competitive ratio of $19/27 \approx 0.704$. To the best of our knowledge, this is the first bound beating $1 - 1/e \approx 0.63$ in the K-stage bipartite matching problems (weighted or unweighted) for any $K > 2$. While our results pertains to fractional allocations, they can easily be extended to the integral version under the large budget assumption. In fact, in the multi-stage configuration allocation problem (and all of its special cases) one can use a simple independent rounding schemes after slightly decreasing the capacities, and show the resulting multi-stage integral algorithm has a competitive ratio no smaller than $\Gamma(K) - O\left(\sqrt{\log(B_{\min})/B_{\min}}\right)$ using standard concentration bounds (e.g., see Devanur et al. (2011) for technical details). Here, $B_{\min}$ should be interpreted as the minimum capacity/required number of impressions of each offline node/advertiser.

Overview of the Main Technique Considering the multi-stage vertex weighted matching problem first, any greedy algorithm allocates the arriving batch of demand requests to the available pool of offline supply at each stage $k \in [K]$, so as to maximize the matching weight of the current stage. Such an

[7] In fact, a simple asymptotic expansion shows that $\Gamma(K) = \left(1 - \frac{1}{e}\right) + \frac{1}{2eK} + O\left(\frac{1}{K^2}\right)$.

allocation possibly suffers from the lack of enough *balancedness* on the supply side — a property that can help the algorithm to remain competitive against the adversary in future stages. To fix this issue, we replace greedy with a more balanced allocation that (i) takes into account the matching weight of each stage, and (ii) simultaneously provides *hedging* against future stages in a controlled fashion. For the special case of online arrivals, this simple idea is the essence of the celebrated BALANCE algorithm (Aggarwal et al., 2011), which is optimal competitive under online arrivals.

To systematically extend the above balanced matching idea under the online setting to the multi-stage setting — where the algorithm can basically match multiple vertices of the arriving batch to the offline side — we propose using specific *stage-dependent convex programs* and output their optimal solution in each stage as the fractional matching of that stage. Given the vector of remaining capacities at the beginning of each stage, the convex program for this stage essentially finds a feasible fractional allocation that maximizes a regularized version of the total weight objective function. In particular, in order to favor more balanced fractional allocations, we choose our regularizers to be *specific* (strictly) concave functions of the final matching degrees (or allocation levels) of the offline vertices. More precisely, we use the primitive function $F_k(x) \triangleq \int_0^x f_k(y)dy$ to map the normalized fractional degree (or normalized allocation level) of each offline vertex to a penalty term in $[0,1]$ at stage k. We then subtract the weighted sum of these penalties from the total weight objective function to construct our regularized objective function in that stage; see the convex program $\mathcal{P}_k^{\text{VWM}}$ used in Algorithm 1.

Considering the above algorithmic construct, one major question is how to pick these strictly concave regularizers to achieve our desired $\Gamma(K)$ competitive ratio in K stages. Our intuition suggests that an appropriate multi-stage algorithm requires hedging more in earlier stages and less in later stages, e.g., in the extreme case of the last stage, picking a maximum weight matching is optimal and no balancedness is required. To obtain the desired variation in the balancedness of our allocation, i.e., having a more balanced allocation early on and gradually pushing it towards a greedy allocation in later stages, we aim to incorporate "more intense" regularization at first and then gradually shift towards no regularization in the course of K stages.

Our key technical ingredient that formalizes the above intuitive notion of regularization intensity is introducing a particular sequence $f_1, f_2, \ldots, f_K : [0,1] \to [0,1]$ of *polynomials of decreasing degrees*, where f_k is a polynomial of degree $(K-k)$ associated to stage $k \in [K]$. See Figure 2 and Proposition 1. Here, the regularization intensity of each polynomial is essentially governed by its degree — hence our sequences starts with the highest degree and goes in the descending order of polynomial degrees. We identify this sequence using a backward recursive equation (will be detailed later). We then show this sequence *exactly* characterizes the regularizers of the optimal competitive multi-stage algorithm. Interestingly, the first-stage polynomial f_1 converges to $f(x) = e^{x-1}$ as K goes to infinity. Also, in the special case of batches of size one, replacing the polynomial f_k at all stages with *the same function*

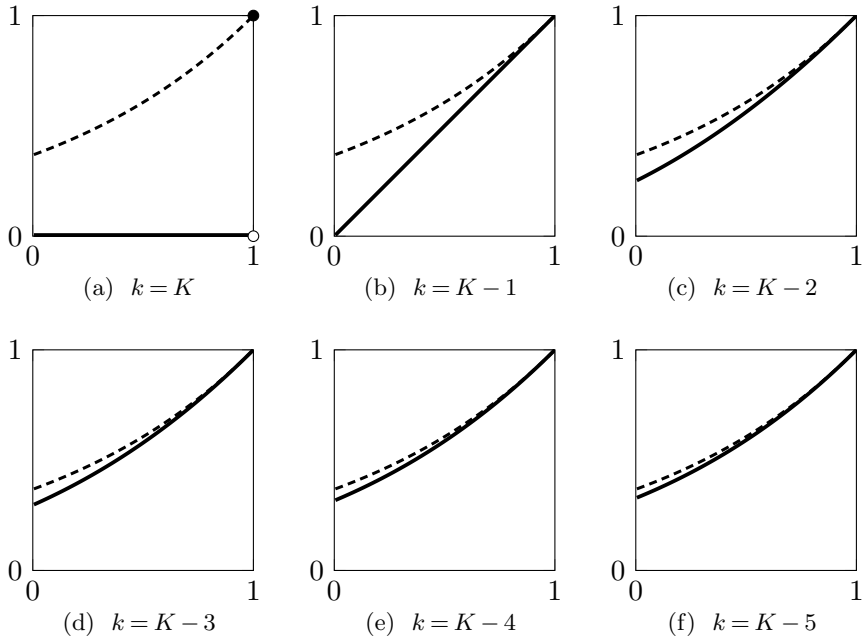


Figure 2 **Polynomial regularizer $f_k(x)$ in Proposition 1 (solid line) versus e^{x-1} (dashed line).**

$f(x) = e^{x-1}$ recovers the $(1-1/e)$-competitive BALANCE algorithms in Aggarwal et al. (2011) for the online vertex weighted b-matching.

To analyze our polynomial regularized matching algorithms — inspired by the technique introduced in Feng et al. (2020b) for two-stage matching — we use the strong duality of the convex programs, tied to a specific primal-dual analysis that is quite different from the known primal-dual analysis in the literature for various forms of online bipartite allocation problems. The sketch of our analysis is roughly the following: The Karush–Kuhn–Tucker (KKT) conditions at each stage k offer a structural decomposition of the subgraph induced by the k^{th} batch. The special unweighted case of this decomposition coincides with the well-studied *matching skeleton* introduced by Goel et al. (2012), which itself is closely related to the Edmonds-Gallai decomposition (Edmonds, 1965; Gallai, 1964) of bipartite graphs. Using this decomposition, and by carefully using our polynomials $f_1, \ldots, f_k$, we propose a novel *recursive primal-dual framework* to bound the competitive ratio of our fractional matchings. The resulting dual constructions, together with our recursive formulation, shows how to improve the competitive ratios from $(1-1/e)$ to $\Gamma(K)$ in the multi-stage setting with K batches. In other words, this new algorithmic construct extends the dual constructions in previous work (Mehta et al., 2007; Aggarwal et al., 2011; Buchbinder et al., 2007) in a novel fashion to the multi-stage setting with batch arrivals. As a side dish, our technique provides a convex-programming based interpretation for the existing primal-dual proofs in the special case of the online setting.

New Challenges for the Configuration Allocation Switching to the multi-stage configuration allocation problem with free disposals, we aim to extend our earlier algorithm to this model; nevertheless, as it turns out, our earlier algorithmic technique cannot be applied directly to the multi-stage

configuration allocation problem because of three new fundamental technical challenges: (i) *Price-level treatment:* each advertiser pays for impressions at different price levels depending on the selected configuration, which is in a stark contrast to vertex-weighted matching where all impressions are essentially sold at the same price. Thus, it is not clear how to define a regularizer term to mimic the earlier approach, (ii) *Lack of skeleton structure:* moving from matching to configuration allocation breaks several of the structural properties of the problem, most importantly the existence of a matching skeleton graph decomposition as mentioned earlier, which is crucial in the analysis of the multi-stage vertex-weighted matching, and (iii) *Managing preemption:* as the algorithm decides on the choice of the configuration at each time, it implicitly decides also on the number of impressions that each advertiser is responsible to pay at each price level at this point (and basically the rest are preempted); these price-level dependent preemption or free-disposal decisions regarding previous impression-allocations are crucial to be able to obtain any bounded competitive algorithm, and a priori it is not clear how to extend our previous approach to help with these preemption decisions.

The first new key technical ingredient to circumvent the above challenges is the stage-dependent *price-level regularized convex program* $\mathcal{P}_k^{\text{MCA}}$ used by Algorithm 2 in each stage $k \in [K]$. Similar to our previous convex programs, this new program maximizes the total revenue minus a regularization term by fractionally assigning a configuration to each user in batch k and deciding which impressions should be preempted by each advertiser during stage k. This time, the regularization term is a summation of several terms, where we have one term for each advertiser at *each possible per-impression price level*. More precisely, we keep track of the price distribution of top impressions allocated to each advertiser given her capacity (i.e., the number of impressions at different per-impression price levels that the advertiser is willing to pay so far), instead of just her remaining capacity. Having this distribution at the beginning of each stage k as its input, the regularizer term corresponding to this advertiser is a separable function across all the price levels, where each term applies the primitive polynomial $F_k(x)$ in Proposition 1 to map the (normalized) number of allocated impressions at some price level to a penalty term in the objective. Intuitively speaking, this new form of regularization helps with gradually decreasing the hedging for each price level separately across stages.

The second new key technical ingredient is how to connect convex programs $\mathcal{P}_k^{\text{MCA}}$ in each stage to an LP primal-dual analysis for the configuration allocation. As mentioned earlier, in contrast to the vertex-weighted matching, in configuration allocation the resulting convex programs *do not admit* any matching skeleton or similar graph decompositions. The absence of this combinatorial structure, which has been the backbone of our previous primal-dual analysis, combined with price-level regularization and the need to incorporate preemption decisions in the analysis, make this new analysis more challenging. We circumvent this last obstacle by considering the Lagrangian dual program $\mathcal{D}_k^{\text{MCA}}$ of the convex program $\mathcal{P}_k^{\text{MCA}}$, and *directly* using its optimal dual solution to construct the dual assignment in

the primal-dual analysis framework. Surprisingly, even with this different proposed dual assignment, we can connect the analysis of the competitive ratio to the same recursive functional equation across K stages as before, and thus obtaining the $\Gamma(K)$ competitive ratio by Algorithm 2. We should highlight that the connection between LP primal-dual analysis and the convex duality in our paper might be of independent interest in other online algorithm design problems.

We finally show the competitive ratio $\Gamma(K)$ is the best achievable by any fractional multi-stage algorithm, whether deterministic or randomized, by showing an upper-bound instance for the special case of the unweighted multi-stage matching. Notably, this instance relies on batches of size more than one, and hence cannot be used as an upper-bound instance for the special case of online arrival where batch sizes are all one. See Figure 4 for more details.

Managerial Insights Our study sheds insights on the role of batching in service operation of matching platforms, and provides algorithmic evidences on how they should adaptively tune their batched matching algorithms to achieve the right trade-off between *"greedy-ness"* in the current decision and "hedging" for the sake of future decisions in a prior-free non-stationary environment:[8]

- In settings that allow for latency—and hence allow for batching the demand arrivals—batched matching algorithms (versus fully online algorithms) can be used by the algorithm designer to obtain performance improvements, even when the environment is highly non-stationary.
- There is an inherent trade-off between the gain from batching (which is captured by the competitive ratio in our competitive analysis framework) and the degree of batching (which is captured by the inverse number of batches in our framework) in prior-free non-stationary dynamic allocations: with less number of batches/stages—which accounts for a higher degree of batching—a well-designed multi-stage matching algorithm can obtain a higher performance; moreover, it gets closer to the offline optimal matching as the degree of batching increases.[9]
- When designing batched matching algorithms, our result suggests that pushing the matching to be more robust to future uncertainty early on (i.e., to be more balanced) and then pushing it to be more greedy later on will help with the performance in a finite horizon problem. In particular, we could formally capture this design aspect using our decreasing degree polynomial regularizers; However, in general, decreasing the degree of hedging induced by the matching algorithm over time to obtain improved performance is the main operational takeaway message of our work, and there might be other ways of performing this task in practice.

Organization In Section 2, we start by formalizing both our baseline model (Section 2.1) and extension model (Section 2.2). In Section 3, we introduce our first algorithm by starting with some

[8] We would like to reiterate that in our paper we make no assumptions on the number of nodes in each batch.

[9] Of course, there is no free lunch here and the actual cost of batching is the induced latency in decision making.

illustrative examples (Section 3.1). We then present this optimal competitive multi-stage algorithm for the vertex weighted bipartite matching problem (Section 3.2), and analyze its competitive ratio lower bound (Section 3.3). In Section 4, we switch to our extension model. We present our second algorithm, which is an optimal competitive multi-stage algorithm for the configuration allocation problem (Section 4.1), and characterize some of its properties (Section 4.2). We then analyze its competitive ratio lower bound (Section 4.3). We then show that our competitive ratio lower-bounds *exactly* match the upper-bound on the competitive ratio of *any* multi-stage algorithm (Section 5). We finally numerically evaluate the performance of our multi-stage algorithms on synthetic instances (Appendix D), and also discuss challenges one might face in order to use algorithmic insights developed in our work in practical applications that allow for batching (Appendix F).

1.1. Further Related Literature

Here is a summary of the other work in the literature directly related to our work.

Multi-stage matching. Our work is closely related to Feng et al. (2020b), which studies the optimal competitive two-stage vertex weighted matching and pricing under both adversarial and stochastic settings. We extend their result in the adversarial case to multi stages and settings beyond vertex weighted matching (in particular to multi-stage edge-weighted matching and configuration allocation with free disposal settings). In Feng et al. (2020b) basically there is a large class of convex programs that can be used in the first stage to introduce hedging. In our paper, we diverge by considering a *different* form of convex-programming based matching where we use regularizers, and we show by changing the regularizer at each stage the optimal competitive ratio can be achieved for any K. Notably, both papers use a primal-dual analysis to analyze the underlying algorithm, but our dual construction in the analysis follows a different logic and has a different algebraic form (even for the special case of the two-stage problem). Our result for the unweighted multi-stage matching is closely related to the beautiful work of Lee and Singla (2017), which also uses the matching skeleton of Goel et al. (2012), but for a different model of arrivals and online problem.

Two-stage stochastic programming. Two-stage stochastic combinatorial optimization problems have been studied extensively in the literature (e.g., Birge and Louveaux, 2011; Charikar et al., 2005; Shmoys and Swamy, 2004; Shmoys and Sozio, 2007; Immorlica et al., 2004; Hanasusanto et al., 2015; Hanasusanto and Kuhn, 2018). Also, see the excellent book of Kuhn (2006) for more context regarding convex multi-stage stochastic programs. For two-stage stochastic matching, various models with different objectives have been studied in Kong and Schaefer (2006); Escoffier et al. (2010); Katriel et al. (2008) and more recently in Feng et al. (2020b). Another work that is conceptually related to us is the recent work of Housni et al. (2020), which is inspired by the idea of robust multi-stage optimization (e.g., Bertsimas et al., 2010, 2011). They consider a two-stage robust optimization for

cost-minimization of the matching, where future demand uncertainty is modeled using a set of demand scenarios (specified explicitly or implicitly). Our model and results diverge drastically from this work.

Online bipartite allocations. Our results are comparable to the literature on online bipartite matching (with vertex arrival) and its extensions. Besides the work mentioned earlier, related to us is Birnbaum and Mathieu (2008), that gives a simple non primal-dual proof of RANKING algorithm of Karp et al. (1990) for online bipartite matching. An elegant primal-dual analysis is discovered later in Devanur et al. (2013). Also, at high-level, our primal-dual analysis resembles some aspects of Buchbinder et al. (2007); Feldman et al. (2009a); Devanur and Hayes (2009); Devanur and Jain (2012); Esfandiari et al. (2015); Huang et al. (2018); Feng et al. (2019). Other related models are online bipartite stochastic matching (e.g., Feldman et al., 2009b; Bahmani and Kapralov, 2010; Manshadi et al., 2012; Mehta et al., 2014; Jaillet and Lu, 2014) and online bipartite matching with stochastic rewards (e.g., Mehta and Panigrahi, 2012; Goyal and Udwani, 2020; Huang and Zhang, 2020). We refer interested readers to a comprehensive survey by Mehta (2013) for further related work. Finally, related to us is also the literature on stochastic online packing and convex programming with large budgets (e.g., Feldman et al., 2010; Devanur et al., 2011; Agrawal and Devanur, 2014). In particular, the role of concave regularizers have been studied in these settings for different purposes, e.g., fairness (Agrawal et al., 2018; Balseiro et al., 2021), robustness to adversarial corruptions (Balseiro et al., 2022), and obtaining blackbox reductions from Bayesian mechanism design to algorithm design (Dughmi et al., 2021).

Applications in revenue management and service operations. From both methodological and application points of view, our work is related to several streams of work that focus to design online allocations and matching algorithms for problems in revenue management and service operations. These work pertain to practical applications in online retail, ride-sharing and other online platforms. Some examples are online assortment optimization (e.g., Golrezaei et al., 2014; Ma and Simchi-Levi, 2020), online assortment of reusable resources (e.g., Gong et al., 2022; Feng et al., 2019; Goyal et al., 2020), Bayesian online assortment with heterogeneous customers (e.g., Rusmevichientong et al., 2020; Feng et al., 2020a), online assortment with replenishment (Feng et al., 2021), dynamic pricing with static calendar (Ma et al., 2021), volunteer crowd-sourcing and nudging (Manshadi and Rodilitz, 2020), two-sided assortment optimization (Aouad and Saban, 2021), online allocation strategies in two-sided media platforms (Alaei et al., 2022), online allocation of patient demand requests (Golrezaei and Yao, 2021), and finally the line of work on dynamic windowed supply-demand matching/queuing—with applications in ride-sharing (e.g., Ashlagi et al., 2019; Aouad and Saritaç, 2020; Özkan and Ward, 2020; Ata et al., 2020; Aveklouris et al., 2021; Arnosti et al., 2021). Another recent work related to us is Xie et al. (2022) which studies the effect of delay (or equivalently look-ahead) in large-traffic stochastic stationary settings such as the multi-secretary problem (Vera and Banerjee, 2021). Our paper diverges from this work by considering non-stationary and adversarial arrival under batching instead of delay.

Video-ads allocation. The popularity of in-stream video platforms has ignited a new research area related to video-ads, see a recent survey by Frade et al. (2021) for a detailed discussion. There is a limited literature on the algorithmic study of video-ads allocation. Comparing to more traditional applications such as display ads, video-ads (especially in-stream ads) have a distinct feature, which is the need to be shown to the user during the video session. Dasgupta et al. (2009) introduce the "online story scheduling" problem in which the task is to schedule in-stream ads with different time lengths and per-unit prices for a single user in an online fashion, and provide a constant competitive algorithm. Further improvements and studying other variants of this problem have been the focus in the literature, e.g., Albers and Passen (2019); Liu et al. (2016). Backstrom et al. (2009) study an alternative media scheduling problem for multiple users in an offline fashion. Sumita et al. (2017) consider a variant of AdWords problem (Mehta et al., 2007), where each offline node (which is an in-stream ad in their model) is associated with a time length and every online node (i.e., a user) can be matched with multiple offline nodes as long as the total time length is below some threshold.

2. Preliminaries

2.1. Baseline Model: Multi-stage Vertex Weighted Matching

We study *vertex weighted matching* as our baseline bipartite allocation model for matching arriving demand requests to available supply. An instance of our model consists of a bipartite graph $\mathcal{G} = (U, V, E)$, where vertices in V (supply) are present offline and vertices in U (demand) arrive over time. Specifically, we consider a *multi-stage arrival* model, in which vertices in U arrive in $K \in \mathbb{N}$ stages. We assume the number of stages K is fixed and known by the algorithm. At each stage $k \in [K]$, a new subset of online vertices $U_k \subseteq U$ arrive, which we often call an arriving *batch*, and reveal their incident edges in E. We assume $|U_k| \geq 1$ and we make no further assumptions on U_k's. We highlight that $U = \cup_{k\in[K]} U_k$ and $U_k \cap U_{k'} = \emptyset$ for $k \neq k'$. Each offline vertex j has a total capacity n_j for allocations, as well as a non-negative weight w_j for each unit of allocation. After arrival of batch k, the multi-stage allocation algorithm makes an irrevocable allocation decision by allocating each vertex $i \in U_K$ (which corresponds to one unit of demand) to the set of neighboring offline vertices $j \in V, (i, j) \in E$ that have non-zero remaining capacity. Let E_k denote the set of edges in E incident to U_k. We denote the allocation of stage k by the vector $\mathbf{x}_k \in \mathbb{R}^{E_k}$, where $x_{k,ij} \in [0, 1]$ is the amount of allocation of vertex i to j. This allocation can be fractional or integral. We mostly focus on fractional allocations and assume $n_j = 1$ for all the offline vertices $j \in V$ without the loss of generality.[10] The goal of the multi-stage vertex weighted matching algorithm is to maximize the total weight of offline vertices in the (fractional) allocation, while ensuring feasibility, i.e., each online (resp. offline) vertex uses no more than one unit of demand (resp. capacity).

[10] We are essentially considering fractional algorithms and adversarial arrivals; in such a setting, without the loss of generality, we can focus on adversaries that choose the input instance adaptively.

2.2. Extension Model: Multi-stage Configuration Allocation

Motivated by the application of displaying video-ads in video streaming platforms, we also study the extension of our baseline model to *multi-stage configuration allocation*. An instance of this model consists of a set of offline advertisers V who are willing to pay the platform to display their ads to its users and a set of arriving users U who are planning to watch videos on this platform. Specifically, the platform needs to choose an advertising *configuration* $c \in C$ for each user (or equivalently, for each video session watched by a user), where C is the set of feasible configurations.[11] Fixing a user $i \in U$, choosing each configuration $c \in C$ allocates a certain number of reserved impressions $\xi_{i,c,j}$ at a non-negative per-impression price $w_{i,c,j}$ to each advertiser $j \in V$. Each advertiser j demands n_j number of impressions in total. We assume free-disposal, which means that the total allocated impressions to advertiser j can exceed n_j, but she is eventually only paying for the top n_j most expensive impressions. Similar to our baseline model in Section 2.1, users arrive to the platform over time in $K \in \mathbb{N}$ stages. At each stage $k \in [K]$ a new batch of online users $U_k \subseteq U$ arrives. Upon the arrival of this batch, the video platform observes $\boldsymbol{\xi}^{(k)} = \{\xi_{i,c,j}\}_{i \in U_k, c \in C, j \in V}$ and $\mathbf{w}^{(k)} = \{w_{i,c,j}\}_{i \in U_k, c \in C, j \in V}$. Then the multi-stage allocation algorithm makes an irrevocable decision by assigning a configuration to each user $i \in U_k$. These configuration allocations can be integral or fractional. In this paper we focus on the fractional allocations [12] and assume $n_j = 1$ for all advertisers $j \in V$ without the loss of generality.[13] The goal of the multi-stage configuration allocation algorithm is to choose configurations for the users to maximize the total collected payments from allocating impressions to the advertisers at the end, which we also refer to as the revenue of the platform.

LP Formulation and Offline Benchmark To help with understanding our extension model, we formulate the optimum offline solution in the configuration allocation problem as a simple linear program, denoted by PRIMAL-MCA (which is basically the LP introduced in Devanur et al., 2016):

$$\begin{aligned}
\max_{\mathbf{x},\mathbf{z} \geq \mathbf{0}} \quad & \sum_{i \in U} \sum_{c \in C} \sum_{j \in V} w_{i,c,j} \cdot x_{i,c,j} && \text{s.t.} \\
& \sum_{i \in U} \sum_{c \in C} x_{i,c,j} \leq n_j && j \in V \ , \\
& \sum_{c \in C} z_{i,c} \leq 1 && i \in U \ , \\
& x_{i,c,j} \leq \xi_{i,c,j} \cdot z_{i,c} && i \in U, c \in C, j \in V \ .
\end{aligned} \tag{PRIMAL-MCA}$$

[11] Our model and results allow a different feasible set for each user, but we omit that for simplicity.

[12] Using standard randomized rounding techniques, all of our results for the fractional allocation problem, including the algorithm and its competitive analysis, can easily be extended to the setting with integral randomized allocations under the *large budget assumptions*, i.e, in the asymptotic regime when $\min_{i,c,j} \frac{n_j}{\xi_{i,c,j}}$ is large.

[13] Since we allow fractional solutions, it is without loss of generality to normalize the parameters after a simple change of variables, so that we can assume $n_j = 1$ for all $j \in V$ without loss of generality. We omit details for brevity.

In the above formulation, $z_{i,c}$ is the fraction of configuration c assigned to user i, and $x_{i,c,j}$ is the (fractional) number of impressions created by user i for advertiser j based on configuration c that the advertiser is finally willing to pay. Note that the number $x_{i,c,j}$ of impressions that the advertiser is willing to pay is no more than number of allocated impressions $\xi_{i,c,j} \cdot z_{i,c}$ from user i under configuration c, and the first quantity can be strictly smaller than the second (as advertiser j should only pay for at most n_j impressions overall). If an impression is allocated but the advertiser is not responsible to pay for it, we refer to that impression as being *preempted* or *freely disposed*.

We finally note that the multi-stage configuration allocation problem generalizes the multi-stage vertex weighted matching problem. Also, although we allow for free-disposal in the extension model as allocations of an advertiser's capacity can value differently (i.e., be at a different price level) for different users and configurations, for the special case of vertex weighted matching there is no point in free-disposal by any multi-stage algorithm, as all allocations of an offline node's capacity value the same. See the formal details in Appendix A. This distinction is the key that requires novel extensions of our techniques for the baseline model to capture different price levels.

In Appendix B, we consider other three classic bipartite allocation models (b-matching, AdWords, assortment). By illustrating that all three models are special cases of the configuration allocation problem, we explain how our results apply to these models.

2.3. Competitive Ratio Analysis

We measure the performance of our multi-stage algorithms by their *competitive ratio*, which is the worst-case ratio between the expected total weight (or total revenue) of the algorithm and the optimum offline benchmark given the entire problem instance (formally defined below).

Definition 1 (**Competitive Ratio**). Fix the number of stages $K \in \mathbb{N}$. For $\alpha \in [0,1]$, a multi-stage allocation algorithm $\texttt{ALG}$ is α-*competitive* against the optimum offline solution in the multi-stage vertex weighted matching (resp. multi-stage configuration allocation) problem, if we have:

$$\inf_{I \in \mathcal{I}^{(K)}} \quad \frac{\texttt{ALG}(I)}{\texttt{OPT}(I)} \geq \alpha \ ,$$

where $\mathcal{I}^{(K)}$ is the set of instances of the multi-stage vertex weighted matching (resp. multi-stage configuration allocation) problem with K stages, and $\texttt{ALG}(I)$ and $\texttt{OPT}(I)$ are the total weights (resp. total revenue) of the multi-stage algorithm $\texttt{ALG}$ and optimum offline solution on an instance I, respectively. For randomized multi-stage algorithms, we abuse the notation and $\texttt{ALG}(I)$ denotes the expected total weight (resp. expected total revenue) of the algorithm on an instance I.

2.4. A Recursive Sequence of Polynomials

As a novel technical ingredient of our proposed multi-stage allocation algorithms, we identify a particular sequence of polynomials of decreasing degrees. Our construction of these polynomials is recursive,

which gives us a polynomial of degree $K-k$ for each stage $k=1,2,\ldots,K-1$. Here, we introduce these polynomials, describe this underlying recursive equation, and elaborate a bit more on some of their properties (Proposition 1; see the proof in Appendix C.1). Later, we first show in Section 3 how these polynomials help us to design an optimal competitive multi-stage algorithm for the vertex weighted matching and how to develop a primal-dual analysis framework that incorporates this recursive equation. Then we show how to extend these ideas and use our polynomials for the general multi-stage configuration allocation problem in Section 4, where we require treating allocations at different price levels differently — as will be clear later.

PROPOSITION 1 **(Sequence of Polynomial Regularizers)**. *Consider a sequence of functions* $f_1, f_2, \ldots, f_K : [0,1] \to [0,1]$ *satisfying the following backward recursive equation:*

- $\forall x \in [0,1]:\ f_K(x) = \mathbb{I}\{x=1\}$, *where* $\mathbb{I}(\cdot)$ *is the indicator function,*
- $\forall x \in [0,1], k \in [K-1]:\ f_k(x) = \max\limits_{y \in [0,1-x]} (1-y) f_{k+1}(x+y)$.

Then, the following statements hold:

(i) *The function sequence* $\{f_k\}_{k=1}^{K-1}$ *is a sequence of* $K-1$ *polynomials with decreasing degrees, where* $f_k(x) = (1-\frac{1-x}{K-k})^{K-k}$ *for* $k \in [K-1]$.[14]

(ii) $f_k(x)$ *is monotone increasing and continuous over* $[0,1]$ *for* $k \in [K-1]$. *Hence,* $F_k(x) \triangleq \int_0^x f_k(y)dy$ *is differentiable and strictly convex over* $[0,1]$ *for* $k \in [K-1]$.

(iii) $\forall k \in [K]$, $f_k(x) \le e^{x-1}$ *for* $x \in [0,1]$. *Moreover, the first function* $f_1(x) = (1-\frac{1-x}{K-1})^{K-1}$ *converges point-wise from below to the function* e^{x-1} *as* $K \to +\infty$ *(see Figure 2).*

(iv) $\min\limits_{y \in [0,1]} 1-(1-y)f_1(y) = 1-\left(1-\frac{1}{K}\right)^K \triangleq \Gamma(K)$

The aforementioned polynomials $\{f_k\}_{k \in [K-1]}$ are drawn in Figure 2. As it is clear from this figure, these polynomials provide careful point-wise estimations for the exponential function $f(x) = e^{x-1}$ at different levels of precision. When these polynomials are used to regularize the maximum weight matching (or total revenue) objective functions, they provide different levels of "hedging" intensity against the future uncertainty in the arriving input exactly because of these different levels of estimation precision. This point will be clear in the upcoming sections.

3. Multi-stage Vertex Weighted Matching

In this section we show the result of the baseline model, which is an optimal competitive algorithm for the K-stage (fractional) vertex weighted matching problem. We start by a few simple examples to provide intuitions behind some of the main technical ideas used in the design of our algorithm. Then, in Section 3.2, we formally describe this algorithm and all of its ingredients. We then provide details of our new convex-programming based primal-dual framework in Section 3.3, and show how the recursive sequence of polynomials introduced in Proposition 1 help with analyzing the competitive ratio.

[14] As a convention, we refer to $f_K(x) = \mathbb{I}\{x=1\}$ as a polynomial of degree 0 (although not mathematically rigorous).

3.1. A Few Illustrative Examples

Online Allocation vs. Batched Allocation Consider a simple two-stage matching instance as in Figure 3a, where all the weights w are equal to 1. Now, consider the simple fractional allocation algorithm BALANCE, also known as *water-filling*, that is commonly used in the literature of online bipartite matching (Mehta et al., 2007; Aggarwal et al., 2011). Upon arrival of batch U_1, BALANCE goes over demand nodes in U_1 in some order and allocate each $i \in U_1$ continuously and fractionally to the supply node in V that has the largest remaining capacity. As a result, no matter what ordering it picks, the algorithm produces the allocation vector $(0.25, 0.25, 0.25, 0.25)$ for u_1^1 and $(1/3, 1/3, 1/3)$ for u_1^2, as it can be seen in Figure 3a. Fixing the first stage allocation, it is easy to check that in the worst-case (for the competitive ratio) the adversary will add two edges in the second stage to two of the supply nodes with the smallest remaining capacity, namely (u_2^1, v_3) and (u_2^2, v_4). In this case, the algorithm obtains a total objective value of $2 + 2 \times (1 - 0.25 - 1/3) \approx 2.83$, while the optimum offline has an objective value of 4; hence a ratio of $2.83/4 < 0.75$. Alternatively, consider a *batched water-filling* allocation, where we try to fractionally allocate demand nodes in U_1 simultaneously to supply nodes V, so that the remaining capacities of supply nodes will be as balanced as possible. This algorithm produces the allocation vector $(0.5, 0, 0.5, 0)$ for u_1^1 and $(0.5, 0, 0.5)$ for u_1^2, as it can be seen in Figure 3b. The same second stage would be the worst-case for competitive ratio of this algorithm, under which the algorithm obtains a total objective value of $2 + 1 = 3$, while the optimum offline obtains 4; hence a ratio of 0.75. As it can be seen from this example, the batched allocation could obtain the (optimal) ratio of 0.75, while the algorithm that mimicked the online allocation could only obtain a ratio strictly smaller than 0.75.

Naive vs. Weight-dependent Batched Allocations Consider a slightly different instance as in Figure 3c, in which we remove the edge (u_1^2, v_3) in the first stage and supply nodes either have weight w or $\hat{w}$, and $w \gg \hat{w}$. A naive batched water-filling allocation as before will obtain an objective value of $0.5 \times (2w + 2\hat{w}) + \hat{w} = w + 2\hat{w}$, while the optimum offline has an objective value of $2w + 2\hat{w}$; hence a ratio of $\frac{w+2\hat{w}}{2w+2\hat{w}} \approx 0.5$. Now consider a different batched allocation algorithm that does take weights into account: it finds a fractional allocation that maximizes $\sum_j w_j y_j - \frac{1}{2}\sum_j w_j y_j^2$, where y_j is the final allocation amount of supply node j after the first stage. Such a fractional matching can be found by solving a concave program, and it is easy to check that it produces the allocation vector $(\frac{w}{w+\hat{w}}, 0, \frac{\hat{w}}{w+\hat{w}}, 0)$ for u_1^1 and $(\frac{w}{w+\hat{w}}, \frac{\hat{w}}{w+\hat{w}})$ for u_1^2, as it can be seen in Figure 3d. The same second stage graph would also be the worst-case for competitive ratio of this algorithm. In that case, the algorithm obtains an objective value of

$$2w \times \frac{w}{w+\hat{w}} + 2\hat{w} = \frac{2w^2 + 2\hat{w}^2 + 2w\hat{w}}{w+\hat{w}},$$

while the optimum offline obtains an objective value of $2w + 2\hat{w}$. Considering the ratio, we have

$$\frac{\frac{2w^2+2\hat{w}^2+2w\hat{w}}{w+\hat{w}}}{2w+2\hat{w}} = \frac{w^2+\hat{w}^2+w\hat{w}}{w^2+\hat{w}^2+2w\hat{w}} \geq \frac{3}{4},$$

simply because $4w^2 + 4\hat{w}^2 + 4w\hat{w} - 3w^2 - 3\hat{w}^2 - 6w\hat{w} = (w - \hat{w})^2 \geq 0$. Note that equality holds if and only if $w = \hat{w}$.

The above examples highlight that (i) in a multi-stage bipartite allocation problem, batched allocation can potentially help with improving the competitive ratio, and (ii) a batched allocation algorithm can benefit from using weights in a structured fashion. Finally, the second example suggests that using regularizers for the maximum weight allocation problem is a promising approach to induce more balanced-ness to our allocation algorithm at each stage. This investigation leaves us with the following question: *(a) how to use regularizers in a systematic fashion, to capture the information about the weights and the revealed subgraph at each stage?* Also, intuitively speaking, one might expect that the batched allocation algorithm benefits from having a more balanced allocation early on, to hedge against the uncertainty in future stages, and being more greedy later. Now we can ask: *(b) how to gradually change the balanced-ness of the allocation across different stages, so that we hedge more aggressively early on and less aggressively towards later stages?* As we will see in the next subsection, we address both of these questions by using the sequences of polynomials introduced in Section 2.4 to regularize the stage-wise maximum weight allocation linear programs.

3.2. Polynomial Regularized Maximum Weight Matching

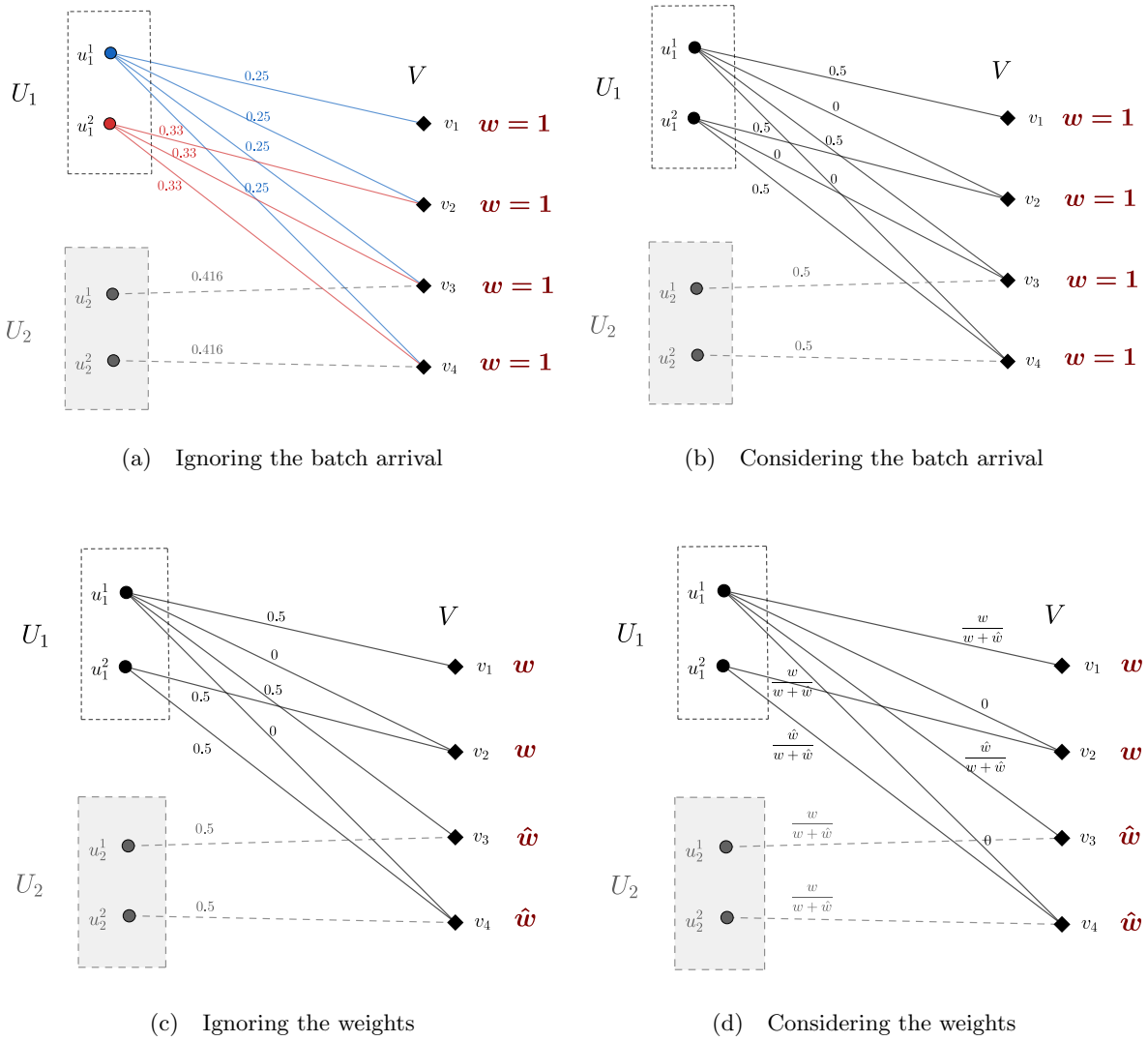
Recall the sequence of polynomials $f_1, f_2, \ldots, f_K$ defined in Section 2 (see Proposition 1 and Figure 2 for details). We are now ready to describe our main algorithm, which is an optimal competitive multi-stage algorithm for vertex weighted bipartite matching.

Overview of the Algorithm Given the sequence of polynomials in Proposition 1, our algorithm solves a different convex program at each stage $k \in [K]$, and returns its unique optimal solution as the fractional matching of that stage. The program of stage k is essentially a regularized maximum vertex weighted matching considering the current remaining capacity of each offline vertex. In this convex program, the regularization is done by a weighted combination of polynomials $F_k(x) = \int_0^x f_k(y)dy$ over offline vertices, each evaluated at the fractional degree of the offline vertex at the end of the allocation. Notably, the convex program at stage k depends on the current state of the allocations, which is captured by the vector of fractional degrees of each offline vertex j at the beginning of stage k denoted by $\mathbf{y}^{(k)}$. See Algorithm 1 for a formal description.

We are now ready to describe the main result of this section.

THEOREM 1 **(Optimal Multi-Stage Algorithm for Weighted Bipartite Matching)**. *In the K-stage (fractional) vertex weighted matching problem, `PR-MWM` (Algorithm 1) is $\Gamma(K)$-competitive, where $\Gamma(K) = 1 - \left(1 - \frac{1}{K}\right)^K$.*

High-level Proof Overview In a nutshell, we analyze our convex programming based algorithm by following two steps. First, we apply the theory of convex duality to the convex programs corresponding



(a) Ignoring the batch arrival

(b) Considering the batch arrival

(c) Ignoring the weights

(d) Considering the weights

Figure 3 Four simple examples of multi-stage vertex-weighted matching; We consider two different instances of this problem (one instance for (a) and (b), and one for (c) and (d)); Both problem instances have two stages, where $U_1 = \{u_1^2, u_1^2\}$, $U_2 = \{u_2^2, u_2^2\}$ and $V = \{v_1, v_2, v_3, v_4\}$; edges are as drawn in the figures above; in (a) and (b) all the weights w_1, w_2, w_3 and w_4 are equal to 1; in (c) and (d), $w_1 = w_2 = w$ and $w_3 = w_4 = \hat{w}$; We run online water-filling, batched water-filling, batched water-filling (ignoring weights), and convex-programming based batched matching (considering the weights) algorithms in (a), (b), (c) and (d), respectively.

to each stage, which guides us to build a convex program specific graph decomposition for each stage's subgraph. Intuitively speaking, the graph decomposition in each stage partitions the subgraph of that stage into groups of demand and supply nodes, so that in each group there exists a fully "weighted balanced" fractional matching (measured by some definition related to the convex program). Second, we try to find an assignment to the dual linear program of our optimum offline LP that is built stage by stage, and in each stage uses the graph decomposition of that stage. We suggest a particular

Algorithm 1: Polynomial Regularized Max Weight Matching (`PR-MWM`)

Input: Number of batches/stages K

$\forall j \in V:\ y_j^{(0)} \leftarrow 0$.

for *stage* $k=1$ *to* K **do**

`/*` k^{th} `batch of vertices in` U`, denoted by` U_k`, arrives. Let` E_k `denote the set of edges incident to vertices in` U_k`. */`

Given $\mathbf{y}^{(k-1)}$ and subgraph $G_k=(U_k,V,E_k)$, solve the convex program $\mathcal{P}_k^{\text{VWM}}$:

$$\mathbf{x}^{(k)}=\arg\max_{\mathbf{x}\geq\mathbf{0}} \sum_{(i,j)\in E_k} w_j x_{ij} - \sum_{j\in V} w_j F_k\big(y_j^{(k-1)}+\sum_{i:(i,j)\in E_k} x_{ij}\big) \text{ s.t.}$$

$$\begin{aligned} \sum_{j:(i,j)\in E_k} x_{ij} &\leq 1 && i\in U_k \\ \sum_{i:(i,j)\in E_k} x_{ij} &\leq 1-y_j^{(k-1)} && j\in V \end{aligned} \qquad (\mathcal{P}_k^{\text{VWM}})$$

`/* Recall that the` k^{th} `regularizer is defined as` $F_k(x)\triangleq\int_0^x f_k(y)dy$`. */`

$\forall j\in V: y_j^{(k)} \leftarrow y_j^{(k-1)}+\sum_{i\in U_k} x_{ij}^{(k)}$.

Return $\mathbf{x}^{(k)}$ as the fractional matching of stage k.

graph-decomposition based dual assignment, where we use the same assignment for all supply nodes in the same group and the same assignment for all demand nodes in the same group. Further, we show the dual objective value is equal to the primal algorithm's objective value from our construction. We finally show this dual assignment is approximately feasible, where the approximation factor is $\Gamma(K)$. It turns out that the approximate feasibility of our dual assignment boils down to the requirement that functions $f_1, f_2, \ldots, f_K$ satisfy the specific recursive equation in the second bullet of Proposition 1 for $k=1,2,\ldots,K-1$. Hence, our `PR-MWM` algorithm (Algorithm 1) obtains the desired competitive ratio of $\Gamma(K)$, as it uses the sequence of polynomials of decreasing degrees in place of $f_1,\ldots,f_K$ that are the unique solution to mentioned recursive functional equation. See the details of this proof overview in Section 3.3.

Remark 1 (**Batched Water-filling**). For the special case of unweighted matching, it is not hard to see that the solution of the convex program $\mathcal{P}_k^{\text{VWM}}$ does *not* depend on the choice of the regularizer function F_k (it only needs to be a strictly convex function). The algorithm for this special case is indeed the batched water-filling algorithm that we investigated in our first example in Section 3.1. This algorithm is oblivious to the choice of the regularizer and does not even need to know K. Note that while in that simple example the batched water-filling ended up with fully balancing the remaining capacities of all the offline vertices in the first stage graph, this is not going to be necessarily the case

in an arbitrary graph at any stage. We should also highlight that this algorithm generalizes the classic water-filling/BALANCE algorithm for the online bipartite fractional matching (Mehta et al., 2007; Devanur et al., 2013) to the setting where we have batches. Importantly, even though our polynomials are not *explicitly* used by the algorithm, they still help with its competitive ratio analysis and showing that it is $\Gamma(K)$-competitive.

REMARK 2 (**Vertex-weighted Online Bipartite Matching**). Another important special case is when batches have size one, which is basically the online arrival problem. As mentioned earlier, by replacing $f_k(x)$ with e^{x-1} in *every stage*, the final allocation of our algorithm will be the same as the classic BALANCE algorithm for the fractional vertex weighted online matching (Aggarwal et al., 2011; Devanur et al., 2013). In other words, the special case of our algorithm provides a convex programming based interpretations of the BALANCE algorithm.[15]

REMARK 3 (**Unweighted Online Bipartite Matching**). Our result implies the following simple corollary for the very special case of the classic online bipartite matching problem: the water-filling/BALANCE algorithm is $\Gamma(n)$ competitive for the (unweighted) online bipartite matching problem with a fixed number of n online vertices. Interestingly, this corollary was also observed in Goel and Mehta (2008), but for the RANKING algorithm (Karp et al., 1990) instead of water-filling, and through a more refined combinatorial analysis of RANKING using a factor revealing LP. As we will see later in Section 5 and Remark 5, this bound is not necessarily tight for the online bipartite matching problem with n online nodes—despite the fact that the competitive ratio $\Gamma(K)$ is indeed optimal in multi-stage matching with K stages (Theorem 3).

3.3. Recursive Primal-dual Analysis of Algorithm 1

We prove Theorem 1 by a primal-dual argument that carefully uses our polynomials. Consider the offline primal LP of the maximum vertex weighted matching in graph $G=(U,V,E)$, and its dual:

$$\max_{\mathbf{x}\geq\mathbf{0}} \quad \sum_{(i,j)\in E} w_j x_{ij} \quad \text{s.t.} \qquad \min_{\boldsymbol{\alpha},\boldsymbol{\beta}\geq\mathbf{0}} \quad \sum_{i\in U}\alpha_i+\sum_{j\in V}\beta_j \quad \text{s.t.}$$
$$\sum_{j:(i,j)\in E} x_{ij}\leq 1 \quad i\in U\ , \qquad \alpha_i+\beta_j\geq w_j \quad (i,j)\in E\ , \qquad \text{(PRIMAL-DUAL-MWM)}$$
$$\sum_{i:(i,j)\in E} x_{ij}\leq 1 \quad j\in V\ ,$$

Given the feasible primal assignment $\{\mathbf{x}^{(k)}\}_{k\in[K]}$ produced by Algorithm 1, we construct a dual assignment so that,

- Primal and dual have the same objective values: $\sum_{k\in[K]}\sum_{(i,j)\in E_k} w_j x_{ij}^{(k)}=\sum_{i\in U}\alpha_i+\sum_{j\in V}\beta_j$,
- Dual is approximately feasible: $\forall(i,j)\in E:\alpha_i+\beta_j\geq\Gamma(K)\cdot w_j$.

[15] To provide more details, BALANCE can be viewed as running a particular first-order iterative method for solving this convex program (basically running a slight variant of the Frank-Wolfe algorithm (Frank et al., 1956)).

Existence of such a dual assignment will guarantee that primal is $\Gamma(K)$-competitive.

Our dual assignment is constructed stage-by-stage. Moreover, our construction relies on a structural decomposition of the subgraph in each stage k, which itself is tied to the solution of the convex program $\mathcal{P}_k^{\text{VWM}}$ of stage k. We highlight that this decomposition is a slightly modified version of a simpler decomposition first introduced in Feng et al. (2020b) (for only the first stage in a two-stage problem and for a slightly different convex program) which actually *does not depend* on the choice of the convex function they used. It also extends the well-known matching skeleton for unweighted bipartite graphs (Goel et al., 2012). Notably, in contrast to both of of these decompositions, this new decomposition actually *depends* on the choice of the convex function used in its definition. This is a key technical property used in the analysis of our algorithm. We first define this decomposition in Proposition 2 and show how it can be constructed through convex programming duality (KKT conditions). We then show how it helps us with our dual construction.

PROPOSITION 2 **(Stage-wise Structural Decomposition)**. *Fix any $k \in [K-1]$. Let $\mathbf{x}^{(k)}$ be the optimal solution of the convex program $\mathcal{P}_k^{VWM}$. Consider the subgraph $G'_k = (U_k, V, E'_k)$ of G_k, where $E'_k = \{(i,j) \in E_k : x_{ij}^{(k)} > 0\}$. Let $V_{k,0}$ be the set of offline vertices in V fully matched by $\mathbf{x}^{(k)}$ given the current $\mathbf{y}$, and $U_{k,0}$ be the set of online vertices who are neighbors of $V_{k,0}$ in G'_k, that is,*

$$V_{k,0} \triangleq \{j \in V : y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)} = 1\} \quad \text{and} \quad U_{k,0} \triangleq \{i \in U_k : \exists j \in V_{k,0}, x_{ij}^{(k)} > 0\} \ .$$

Moreover, let the pairs $\{(U_{k,\ell}, V_{k,\ell})\}_{\ell=1}^{L_k}$ identify the L_k connected components of the induced subgraph $G'_k[U_k \setminus U_{k,0}, V \setminus V_{k,0}]$ of G'_k. Then:

1. *Uniformity: $\forall\ \ell \in [0 : L_k], j, j' \in V_{k,\ell}$:*

$$w_j \left(1 - f_k\left(y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)}\right)\right) = w_{j'} \left(1 - f_k\left(y_{j'} + \sum_{i:(i,j')\in E_k} x_{ij'}^{(k)}\right)\right) \triangleq c_\ell^{(k)}$$

2. *Monotonicity: $\forall\ \ell, \ell' \in [0 : L_k]$: there exists no edge in E_k between $U_{k,\ell}$ and $V_{k,\ell'}$ if $c_\ell^{(k)} < c_{\ell'}^{(k)}$.*
3. *Saturation: $\forall\ \ell \in [L_k]$: all online vertices $i \in U_{k,\ell}$ are fully matched by $\mathbf{x}^{(k)}$, i.e., $\sum_{j:(i,j)\in E_k} x_{ij}^{(k)} = 1$.*

We prove the above graph-based structural decomposition by investigating the program $\mathcal{P}_k^{\text{VWM}}$ through convex strong duality. See the details in Appendix C.2. At a high-level, the above graph decomposition partitions the subgraph revealed at each stage k based on a (weighted) notion of *"match-ability"* of the offline nodes (i.e., $c_\ell^{(k)}$ values). Having such a partitioning helps us to identify a charging scheme, so that we can charge the primal contributions of our algorithm between the dual variables corresponding to offline and online nodes appropriately. We now have all the ingredients to prove our main theorem of this section using a primal-dual analysis.

Proof of Theorem 1. For each stage $k \in [K-1]$, let $\{(U_{k,0}, V_{k,0}), \ldots, (U_{k,L_k}, V_{k,L_k})\}$ be the structural decomposition of (U_k, V, E_k) as in Proposition 2, and $\{x_{ij}^{(k)}\}_{(i,j)\in E_k}$ be the optimal solution of the program $\mathcal{P}_k^{\text{VWM}}$. Note that if $c_\ell^{(k)} = c_{\ell'}^{(k)}$ for $\ell \neq \ell'$ and some $k \in [K-1]$, we can simply merge the two pairs $(U_{k,\ell}, V_{k,\ell})$ and $(U_{k,\ell'}, V_{k,\ell'})$ to $(U_{k,\ell} \cup U_{k,\ell'}, V_{k,\ell} \cup V_{k,\ell'})$, and still our decomposition satisfies the three properties of Proposition 2; these properties are all we need for our technical arguments. Therefore, without loss of generality, we can assume $c_\ell^{(k)}$'s are non-identical. Moreover, $0 = c_0^{(k)} < c_1^{(k)} < \ldots < c_{L_k}^{(k)}$.

First, we define the following notations which will be useful in this proof:

$$\forall j \in V,\ k \in [K]: \qquad y_j^{(k)} \triangleq \sum_{s=1}^{k-1} \sum_{i:(i,j)\in E_s} x_{ij}^{(s)}$$

Now, consider the offline primal-dual linear programs in Primal-Dual-MWM. Given the feasible primal assignment $\{\mathbf{x}^{(k)}\}_{k\in[K]}$, construct a dual assignment as follows. First, set $\alpha_i \leftarrow 0$ and $\beta_j \leftarrow 0$ for all $i \in U, j \in V$. At each stage $k \in [K-1]$, update the dual variables as follows:

$$\begin{aligned}
&\forall i \in U_{k,\ell},\ \ell \in [0:L_k]: && \alpha_i \leftarrow c_\ell^{(k)} \\
&\forall j \in V_{k,\ell},\ \ell \in [0:L_k]: && \beta_j \leftarrow \beta_j + \Delta\beta_j^{(k)} \\
& && \Delta\beta_j^{(k)} \triangleq \left(\sum_{i:(i,j)\in E_k} x_{ij}^{(k)}\right) \cdot \left(w_j - c_\ell^{(k)}\right) \overset{(a)}{=} w_j \left(y_j^{(k+1)} - y_j^{(k)}\right) f_k\left(y_j^{(k+1)}\right)
\end{aligned}$$

where in eq. (a) above we use the uniformity property and the definition of $c_\ell^{(k)}$ in Proposition 2. For the last stage K, note that $F_K(x) = 0$ for all $x \in [0,1]$. Therefore, Algorithm 1 essentially solves the maximum vertex weighted matching in the last stage graph (U_K, V, E_K) where each $j \in V$ has initial capacity $1 - y_j^{(K)}$. Let $\{\hat{\alpha}_i^{(K)}, \hat{\beta}_j^{(K)}\}_{i\in U_K, j\in V}$ be the *optimal* dual solution of this LP. Then update the dual variables as follows:

$$\begin{aligned}
&\forall i \in U_K: && \alpha_i \leftarrow \hat{\alpha}_i^{(K)} \\
&\forall j \in V\ : && \beta_j \leftarrow \beta_j + \Delta\beta_j^{(K)},\ \ \Delta\beta_j^{(K)} \triangleq \left(1 - y_j^{(K)}\right) \hat{\beta}_j^{(K)}
\end{aligned}$$

The rest of the proof is done in two steps:

[*Step i*] *Comparing objective values in primal and dual.* we first show that the objective value of the above dual assignment is equal to the objective value of the primal assignment of Algorithm 1. To show this, we consider each stage k separately. For the last stage K, this holds by construction. For each stage $k \in [K-1]$, note that the primal objective value is decomposed across pairs $(U_{k,\ell}, V_{k,\ell})$, $\ell \in [0:L_k]$, simply because $x_{ij}^{(k)} > 0$ only if i and j belong to the same pair (Proposition 2). As these pairs partition sets of vertices U and V, it is enough to compare the changes in the primal and dual objectives in each pair $(U_{k,\ell}, V_{k,\ell})$ separately. Notice that if $\ell = 0$ then $c_\ell^{(k)} = 0$, and thus,

$$\Delta\left(\text{Dual}_{k,0}\right) \triangleq \sum_{i\in U_{k,0}} \alpha_i + \sum_{j\in V_{k,0}} \Delta\beta_j^{(k)} = \sum_{j\in V_{k,0}} \sum_{i:(i,j)\in E_k} w_j x_{ij}^{(k)} \triangleq \Delta(\text{Primal}_{k,0}).$$

Similarly, for $\ell \in [L_k]$,

$$\begin{aligned}\Delta\left(\text{Dual}_{k,\ell}\right) &\triangleq \sum_{i\in U_{k,\ell}} \alpha_i + \sum_{j\in V_{k,\ell}} \Delta\beta_j^{(k)} = |U_{k,\ell}|\cdot c_\ell^{(k)} + \sum_{j\in V_{k,\ell}} \sum_{i:(i,j)\in E_k} x_{ij}^{(k)} \left(w_j - c_\ell^{(k)}\right) \\ &= |U_{k,\ell}|\cdot c_\ell^{(k)} + \sum_{j\in V_{k,\ell}} \sum_{i:(i,j)\in E_k} w_j x_{ij}^{(k)} - c_\ell^{(k)} \sum_{i\in U_k} \left(\sum_{j:(i,j)\in E_k} x_{ij}^{(k)}\right) \\ &= \sum_{j\in V_{k,\ell}} \sum_{i:(i,j)\in E_k} w_j x_{ij}^{(k)} \triangleq \Delta(\text{Primal}_{k,\ell}),\end{aligned}$$

where the last equality uses the saturation property in Proposition 2 for $l \in [L_k]$.

[*Step ii*] *Checking approximate feasibility of dual.* To show approximate dual feasibility, it is enough to show that for any stage $k \in [K]$ and any arbitrary edge $(i,j) \in E_k$,

$$\alpha_i + \sum_{s=1}^{k} \Delta\beta_j^{(s)} \geq w_j \left(1 - (1-1/K)^K\right) \tag{1}$$

We next show inequality (1) holds for $k \in [K-1]$ by using the following technical lemma, which carefully uses our sequence of polynomial regularizers to establish a lower bound on the LHS of (1).

LEMMA 1. *For any $K \in \mathbb{N}$, $k \in [K-1]$, $x \in [0,1]$, and any $0 = y^{(1)} \leq y^{(2)} \leq \cdots \leq y^{(k)} \leq 1-x$,*

$$(1-x) f_k(y^{(k)} + x) - \sum_{s=1}^{k-1} (y^{(s+1)} - y^{(s)}) f_s\left(y^{(s+1)}\right) \leq 1 - \Gamma(K) = (1-1/K)^K.$$

Now assume $i \in U_{k,\ell}$ and $j \in V_{k,\ell'}$, which implies $c_{\ell'}^{(k)} \leq c_\ell^{(k)}$ (Proposition 2). Then we have:

$$\begin{aligned}\alpha_i + \sum_{s=1}^{k} \Delta\beta_j^{(s)} &= c_\ell^{(k)} + \left(\sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\right)\left(w_j - c_{\ell'}^{(k)}\right) + \sum_{s=1}^{k-1} \Delta\beta_j^{(s)} \\ &\geq c_{\ell'}^{(k)} + \left(\sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\right)\left(w_j - c_{\ell'}^{(k)}\right) + \sum_{s=1}^{k-1} \Delta\beta_j^{(s)} \\ &= w_j \left(1 - f_k\left(y_j^{(k)} + \sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\right) + \left(\sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\right) f_k\left(y_j^{(k)} + \sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\right)\right) \\ &\quad + \sum_{s=1}^{k-1} \Delta\beta_j^{(s)} \\ &= w_j \left(1 - \left(1 - \sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\right) f_k\left(y_j^{(k)} + \sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\right) + \sum_{s=1}^{k-1} \left(y_j^{(s+1)} - y_j^{(s)}\right) f_k\left(y_j^{(s+1)}\right)\right) \\ &\geq w_j \Gamma(K)\ ,\end{aligned}$$

where the first inequality uses $c_{\ell'}^{(k)} \leq c_\ell^{(k)}$, the second equality uses the expression for $c_{\ell'}^{(k)}$ in Proposition 2 due to uniformity, and the last inequality invokes Lemma 1 for the sequence:

$$0 = y_j^{(1)} \leq y_j^{(2)} \leq \ldots \leq y_j^{(k)} \leq 1 - \sum_{i':(i',j)\in E_k} x_{i'j}^{(k)}\ .$$

So far, we have shown (1) holds for all stages $k<K$. For the last stage K, and any edge $(i,j)\in E_K$, notice that $\alpha_i+\Delta\beta_j^{(K)}=\hat{\alpha}_i^{(K)}+(1-y_j^{(K)})\hat{\beta}_j^{(K)}\geq w_j(1-y_j^{(K)})$ by construction. If $y_j^{(K)}=0$, inequality (1) holds. Otherwise, let $k'<K$ be the latest stage before K where there exists $\ell^\dagger$ such that $j\in V_{k',\ell^\dagger}$ and $\Delta\beta_j^{(k')}>0$. By definition, we have $y_j^{(k')}+\sum_{i':(i',j)\in E_{k'}} x_{i'j}^{(k')}=y_j^{(k'+1)}=y_j^{(k)}$. Therefore:

$$\begin{aligned}\alpha_i+\sum_{s=1}^{K}\Delta\beta_j^{(s)}&\geq w_j\left(1-y_j^{(K)}\right)+\sum_{s=1}^{K-1}\Delta\beta_j^{(s)}=w_j\left(1-y_j^{(K)}\right)+\sum_{s=1}^{k'}\Delta\beta_j^{(s)}\\&\geq w_j\left(1-f_{k'}\left(y_j^{(k'+1)}\right)\right)+\sum_{s=1}^{k'}\Delta\beta_j^{(s)}=c_{\ell^\dagger}^{(k')}+\sum_{s=1}^{k'}\Delta\beta_j^{(s)}\\&\geq w_j(1-(1-1/K)^K)\ ,\end{aligned}$$

where the first equality uses the fact $\Delta\beta_j^{(s)}=0$ for all s between $k'+1$ and $K-1$; the second inequality uses the fact that that $f_s(x)\geq x$ for all $x\in[0,1]$ and $s\in[K-1]$ and $y_j^{(k'+1)}=y_j^{(K)}$, the second equality uses the definition of $c_{\ell^\dagger}^{(k')}$; and finally the last inequality holds by exactly the same argument as in the case of $k\in[K-1]$. □

We finish the analysis by proving the technical lemma used in the proof of Theorem 1. It will be further used in the analysis of our multi-stage configuration allocation algorithm in Section 4.

Proof of Lemma 1. We show the statement by induction on k.

Base case ($k=1$). The left hand side of inequality becomes

$$(1-x)f_1(x)\leq 1-\Gamma(K),$$

which holds because of Proposition 1, part (vi).

Inductive step ($1<k\leq K-1$). For any instance $(K,k,x,y^{(1)},\ldots,y^{(k)})$, consider another instance $(K,k-1,x'\triangleq y^{(k)}-y^{(k-1)},y^{(1)},\ldots,y^{(k-1)})$. Note that

$$\begin{aligned}&(1-x)f_k(y^{(k)}+x)-\sum_{s=1}^{k-1}(y^{(s+1)}-y^{(s)})f_s\left(y^{(s+1)}\right)\overset{(a)}{\leq} f_{k-1}(y^{(k)})-\sum_{s=1}^{k-1}(y^{(s+1)}-y^{(s)})f_s\left(y^{(s+1)}\right)\\&=\left(1-(y^{(k)}-y^{(k-1)})\right)f_{k-1}(y^{(k-1)}+(y^{(k)}-y^{(k-1)}))-\sum_{s=1}^{k-2}(y^{(s+1)}-y^{(s)})f_s\left(y^{(s+1)}\right)\\&=(1-x')\,f_{k-1}(y^{(k-1)}+x')-\sum_{s=1}^{k-2}(y^{(s+1)}-y^{(s)})f_s\left(y^{(s+1)}\right)\overset{(b)}{\leq}1-\Gamma(K)\ ,\end{aligned}$$

where ineq. (a) uses Proposition 1 and the recursive definition of $\{f_k(x)\}_{k\in[K]}$, and ineq. (b) uses the induction hypothesis on instance $(K,k-1,x',y^{(1)},\ldots,y^{(k-1)})$, where $x'=y^{(k)}-y^{(k-1)}$. □

4. Extension: Multi-stage Configuration Allocation

In this section, we discuss how our approach can be generalized to the multi-stage configuration allocation problem. There are several key challenges towards this extension. First, in contrast to vertex weighted matching, here each unit of an advertiser's capacity can be allocated at *different prices*. Thus, it is not clear how to use the regularization approach of the previous section. Second, this important distinction also implies that a similar graph-based structural decomposition to analyze a potential convex programming based algorithm might not exist in this model, roughly because the degree of match-ability of an offline node might vary across different price levels. Third, as mentioned earlier in Section 2, the state of allocation in a configuration allocation problem is high-dimensional, as the previous allocations occur at different price levels. Although previous work in the literature on online edge-weighted matching (in particular Devanur et al., 2016) suggests keeping track of the "allocated reward distributions" (which we detail later), it is not clear how to extend our convex program to use this information. Finally, a configuration allocation algorithm should make disposal decisions together with its allocation whenever needed. A priori, it is not clear how to incorporate preemption into our previous convex programs.[16]

We start by introducing a novel algorithmic construct to address the above challenges. We present our new convex-programming based algorithm (Algorithm 2) based on this new technique in Section 4.1. Due to the lack of a structural decomposition, we study the key properties of this convex program directly in Section 4.2 through convex duality. We then try to mimic the recursive primal-dual analysis of Section 3 in Section 4.3, with the major difference that we directly use the optimal Lagrangian dual solution of our convex programs in each stage to find an approximate dual certificate for our primal algorithm in the LP formulation of the offline problem.

4.1. Price-level Regularized Convex Programs and Configuration Allocation

Before describing our algorithm, we make an assumptions for simplicity (without loss of generality) that there exists a finite universe $0 = \hat{w}_0 < \hat{w}_1 < \cdots < \hat{w}_T$ of $T \in \mathbb{N}$ possible per-impression prices. Namely, for each per-impression price $w_{i,c,j}$, there exists an index $\tau \in [T]$ such that $w_{i,c,j} = \hat{w}_\tau$. As we will see soon, our algorithm does not need to know $\{\hat{w}_\tau\}_{\tau\in[T]}$ ahead of time, and its competitive ratio will have no dependency on this finite set.[17]

Overview of the Algorithm At each stage $k \in [K]$, our algorithm solves a stage-dependent convex program that is essentially a regularized version of the LP for a greedy allocation at that stage. Intuitively speaking, the main new ideas behind this convex program are:

[16] Note that edge-weighted online matching is a special case; hence disposals are definitely needed for having a bounded competitive ratio even when there is one offline node.

[17] Notably, the discreetness assumption is mainly for ease of exposition, and our results can immediately be extended to the setting with continuous per-impression prices.

1. Regularizing the revenue function *separately at different price levels* using a convex function in a structured fashion, in order to create a controlled hedging through maintaining balancedness in allocation separately at each price level,
2. Setting it up in a way that we can extract a balanced configuration allocation, together with the necessary amount of preemption at each price level, from the solution of the program.

Formally, we solve the following convex program $\mathcal{P}_k^{\text{MCA}}$ (with variables $\mathbf{x}, \mathbf{y}, \mathbf{z}$) in stage k of our algorithm:

$$\begin{aligned}
\max_{\mathbf{x},\mathbf{y},\mathbf{z}\geq\mathbf{0}} \quad & \sum_{i\in U_k}\sum_{c\in C}\sum_{j\in V} w_{i,c,j}\cdot x_{i,c,j} - \sum_{j\in V}\sum_{\tau\in[T]} \hat{w}_\tau \cdot \left(\eta_{j,\tau}^{(k-1)} - y_{j,\tau}\right) \\
& \quad - \sum_{j\in V}\sum_{\tau\in[T]} (\hat{w}_\tau - \hat{w}_{\tau-1})\cdot F_k\left(\sum_{\substack{i\in U_k, c\in C:\\ w_{i,c,j}\geq\hat{w}_\tau}} x_{i,c,j} + \sum_{\tau'\geq\tau} y_{j,\tau'}\right) \text{ s.t.} \\
& \sum_{i\in U_k}\sum_{c\in C} x_{i,c,j} + \sum_{\tau\in[T]} y_{j,\tau} \leq 1 && j\in V\ , && (\mathcal{P}_k^{\text{MCA}}) \\
& \sum_{c\in C} z_{i,c} \leq 1 && i\in U_k\ , \\
& x_{i,c,j} \leq \xi_{i,c,j}\cdot z_{i,c} && i\in U_k, c\in C, j\in V\ , \\
& y_{j,\tau} \leq \eta_{j,\tau}^{(k-1)} && j\in V, \tau\in[T]\ .
\end{aligned}$$

Table 1 Variables and inputs in program $\mathcal{P}_k^{\text{MCA}}$

variables	$x_{i,c,j}$	fractional number of impressions of user $i\in U_k$ with configuration $c\in C$ to advertiser $j\in V$ that advertiser j is still willing to pay for after stage k
	$y_{j,\tau}$	fractional number of impressions to advertiser $j\in V$ from previous $k-1$ stages that she is still willing to pay for at per-impression price of $\hat{w}_\tau$ after stage k
	$z_{i,c}$	fractional allocation of user $i\in U_k$ with configuration $c\in C$
inputs	$w_{i,c,j}$	per-impression price of user $i\in U_k$ with configuration $c\in C$ to advertiser $j\in V$
	$\xi_{i,c,j}$	reserved impressions of user $i\in U_k$ with configuration $c\in C$ to advertiser $j\in V$
	$\eta_{j,\tau}^{(k-1)}$	fractional number of impressions to advertiser $j\in V$ that she is still willing to pay for at per-impression price of $\hat{w}_\tau$ after stage $k-1$
	$\hat{w}_\tau$	per-impression price indexed by $\tau\in[T]$

In this convex program, the regularization $F_k(x)=\int_0^x f_k(y)dy$'s are computed with the sequence of polynomials $f_1, f_2, \ldots, f_K$ of decreasing degrees introduced in Proposition 1 and used in convex program $\mathcal{P}_k^{\text{VWM}}$ in Section 3. As a reference, we summarize the interpretation of variables $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ and

inputs $(\mathbf{w}^{(k)}, \boldsymbol{\xi}^{(k)}, \boldsymbol{\eta}^{(k)}, \hat{\mathbf{w}})$ of program $\mathcal{P}_k^{\text{MCA}}$ in Table 1. Specifically, this convex program depends on the set U_k of arriving users in stage k, the vector of impression numbers $\boldsymbol{\xi}^{(k)} = \{\xi_{i,c,j}\}_{i \in U_k}$, and the vector of per-impression prices $\mathbf{w}^{(k)} = \{w_{i,c,j}\}_{i \in U_k}$, all of which are revealed to the algorithm at the beginning of stage k. Furthermore, our algorithm keeps track of $\boldsymbol{\eta}^{(k)} = \{\eta_{j,\tau}^{(k)}\}$, where $\eta_{j,\tau}^{(k)}$ is the fractional number of impressions that each advertiser j is willing to pay for each per-impression price τ at the end of each stage k. The vector $\boldsymbol{\eta}^{(k-1)}$, which is basically the price distribution of top impressions allocated to advertiser j with total capacity of $n_j = 1$, captures the state of the allocation at the beginning of stage k. Notably, the above convex program $\mathcal{P}_k^{\text{MCA}}$ depends on this state information $\boldsymbol{\eta}^{(k-1)}$. As for the decision variables, variable $z_{i,c}$ in the above program specifies the fractional allocation of user $i \in U_k$ with configuration $c \in C$. Moreover, variable $x_{i,c,j}$ specifies the fractional number of impressions created by user i form configuration c that advertiser j is willing to pay for after the allocation of stage k, and thus $\xi_{i,c,j} \cdot z_{i,c} - x_{i,c,j}$ impressions of user i for advertiser j are immediately preempted at the end of stage k. Finally, variable $y_{j,\tau}$ specifies the total number of impressions of previous $k-1$ stages that advertiser j is still willing to pay for at per-impression price of $\hat{w}_\tau$ *after* allocation/preemption decisions of stage k. In other words, $\eta_{j,\tau}^{(k-1)} - y_{j,\tau}$ number of impressions of advertiser j which were finalized in the previous $k-1$ stages at per-impression price of $\hat{w}_\tau$ will be preempted after stage k. See Algorithm 2 for a formal description.

Algorithm 2: Polynomial Regularized Max Configuration Allocation (`PR-MCA`)

Input: Number of stages K, convex functions $\{F_k : [0,1] \to [0,1]\}_{k \in [K]}$

$\forall j \in V, \tau \in [T]:\ \eta_{j,\tau}^{(0)} \leftarrow 0.$

for *stage* $k = 1$ *to* K **do**

`/* ` k^{th} `batch of users in` U`, denoted by` U_k`, arrives. */`

Given $\boldsymbol{\eta}^{(k-1)}$, U_k, $\boldsymbol{\xi}^{(k)}$ and $\mathbf{w}^{(k)}$, solve the convex program $\mathcal{P}_k^{\text{MCA}}$ to obtain $(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}, \mathbf{z}^{(k)})$.

$\forall j \in V, \tau \in [T]: \eta_{j,\tau}^{(k)} \leftarrow \sum_{\substack{i \in U_k, c \in C: \\ w_{i,c,j} = \hat{w}_\tau}} x_{i,c,j}^{(k)} + y_{j,\tau}^{(k)}.$

Return $(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}, \mathbf{z}^{(k)})$ as the fractional solution of stage k.

`/*` $\mathbf{z}^{(k)}$`: configuration allocation of stage` k`,` $\boldsymbol{\eta}^{(k-1)} - \mathbf{y}^{(k)}$`: preemption decisions of stage` k`. */`

We are now ready to present the main result of this section. We prove this theorem in Section 4.3.

THEOREM 2 **(Optimal Multi-Stage Algorithm for Configuration Allocation)**. *In the K-stage (fractional) configuration allocation problem, `PR-MCA` (Algorithm 2) is $\Gamma(K)$-competitive, where $\Gamma(K) = 1 - \left(1 - \frac{1}{K}\right)^K$.*

REMARK 4. We note that the multi-stage configuration allocation problem generalizes the multi-stage edge-weighted matching with free-disposal (Feldman et al., 2009a), budgeted allocation or AdWords with no disposal (Mehta et al., 2007), and vertex weighted matching with no disposal (Aggarwal et al., 2011). As we show later in Section 5, no integral or fractional multi-stage algorithm can obtain a competitive ratio better than $\Gamma(K)$ in the multi-stage (unweighted) matching, and hence $\Gamma(K)$ is the optimal competitive ratio for the multi-stage configuration allocation.

4.2. Properties of Price-Level Regularized Convex Programs via Convex Duality

Fix any stage $k \in [K]$. We now aim to identify/characterize some useful properties of the convex program $\mathcal{P}_k^{\text{MCA}}$. To simplify our study, we define the following notation for any $(\mathbf{x}, \mathbf{y})$:

$$H_{j,\tau}(\mathbf{x},\mathbf{y}) \triangleq \sum_{\substack{i\in U_k, c\in C:\\ w_{i,c,j}\geq \hat{w}_\tau}} x_{i,c,j} + \sum_{\tau'\geq\tau} y_{j,\tau'}$$

Note that if $(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}, \mathbf{z}^{(k)})$ is the fractional allocation selected by Algorithm 2 (i.e., optimal solution of $\mathcal{P}_k^{\text{MCA}}$), then by definition, $H_{j,\tau}(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) = \sum_{\tau'\geq\tau} \eta_{j,\tau}^{(k)}$. In other words, $1 - H_{j,\tau}(\mathbf{x}^{(k)}, \mathbf{y}^{(k)})$ is essentially the cumulative per-impression price distribution of advertiser j at the end of stage k. Note that in the first $K-1$ stages, the objective function in program $\mathcal{P}_k^{\text{MCA}}$ is strictly concave, while the convex program degenerates to a linear program in the last stage K. For this reason, we study the properties of stages $k \in [K-1]$ and the last stage K separately.

Properties of Stage $k \in [K-1]$. To characterize the properties of the optimal solution in $\mathcal{P}_k^{\text{MCA}}$ for stage $k \in [K-1]$, we consider its Lagrangian relaxation and KKT optimality conditions. Specifically, the Lagrangian dual $\mathcal{D}_k^{\text{MCA}}$ of $\mathcal{P}_k^{\text{MCA}}$ can be written as follows,

$$\begin{aligned}
\min_{\substack{\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\mu},\boldsymbol{\pi},\\ \boldsymbol{\psi},\boldsymbol{\phi},\boldsymbol{\iota}\geq 0}} \max_{\mathbf{x},\mathbf{y},\mathbf{z}\geq\mathbf{0}} & \sum_{i\in U}\sum_{c\in C}\sum_{j\in V} w_{i,c,j}\cdot x_{i,c,j} - \sum_{j\in V}\sum_{\tau\in[T]} \hat{w}_\tau \cdot \left(\eta_{j,\tau}^{(k-1)} - y_{j,\tau}\right) \\
& - \sum_{j\in V}\sum_{\tau\in[T]} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot F_k\left(H_{j,\tau}(\mathbf{x},\mathbf{y})\right) \\
& - \sum_{j\in V} \theta_j \cdot \left(\sum_{i\in U_k}\sum_{c\in C} x_{i,c,j} + \sum_{\tau\in[T]} y_{j,\tau} - 1\right) - \sum_{i\in U_k} \lambda_i \cdot \left(\sum_{c\in C} z_{i,c} - 1\right) \qquad (\mathcal{D}_k^{\text{MCA}}) \\
& - \sum_{i\in U}\sum_{c\in C}\sum_{j\in V} \mu_{i,c,j}\cdot(x_{i,c,j} - \xi_{i,c,j}\cdot z_{i,c}) - \sum_{j\in V}\sum_{\tau\in[T]} \pi_{j,\tau}\left(y_{j,\tau} - \eta_{j,\tau}^{(k-1)}\right) \\
& + \sum_{i\in U}\sum_{c\in C}\sum_{j\in V} \psi_{i,c,j}\cdot x_{i,c,j} + \sum_{i\in U}\sum_{c\in C} \phi_{i,c}\cdot z_{i,c} + \sum_{j\in V}\sum_{\tau\in[T]} \iota_{j,\tau}\cdot y_{j,\tau}
\end{aligned}$$

We start with the following lemma (proved in Appendix C.3) which builds a connection between the optimal solutions of $\mathcal{P}_k^{\text{MCA}}$ and $\mathcal{D}_k^{\text{MCA}}$. Briefly speaking, this lemma can be thought as a collection of equalities extracted from KKT conditions, which will later be used in the competitive ratio analysis of Algorithm 2.

LEMMA 2 **(Duality-based Connection)**. *Fix any $k \in [K-1]$. Suppose $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ is the optimal solution of the convex program $\mathcal{P}_k^{\text{MCA}}$. There exists non-negative $(\boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\mu}^*, \boldsymbol{\psi}^*, \boldsymbol{\phi}^*)$ such that for all $(i,c,j) \in E_k$*

$$\lambda_i^* \left(\sum_{c \in C} z_{i,c}^* - 1 \right) = 0 \qquad \mu_{i,c,j}^* \left(x_{i,c,j}^* - \xi_{i,c,j} \cdot z_{i,c}^* \right) = 0$$
$$\theta_j^* = 0 \qquad \psi_{i,c,j}^* \cdot x_{i,c,j}^* = 0 \qquad \phi_{i,c}^* \cdot z_{i,c}^* = 0 \qquad \text{(complementary slackness)}$$

$$\sum_{j \in V} \xi_{i,c,j} \cdot \mu_{i,c,j}^* - \lambda_i^* + \phi_{i,c}^* = 0 \qquad \text{(stationarity-1)}$$

$$w_{i,c,j} - \sum_{\tau \in [T]: \hat{w}(\tau) \le w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) - \mu_{i,c,j}^* + \psi_{i,c,j}^* = 0 \qquad \text{(stationarity-2)}$$

Besides the above lemma, we intuitively expects that the optimal solution $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ of $\mathcal{P}_k^{\text{MCA}}$ satisfies the following property: whenever the entire capacity of a given advertiser is exhausted and preemption is needed, it is optimal to dispose impressions which are allocated with the smallest price. This intuition can be formalized below and proved by convex duality (see Appendix C.4).

LEMMA 3 **(Disposing Smallest Price)**. *Fix $k \in [K-1]$. Suppose $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ is the optimal solution of $\mathcal{P}_k^{\text{MCA}}$. For any advertiser $j \in V$, and $\tau \in [T]$, if $\eta_{j,\tau}^{(k-1)} - y_{j,\tau}^* > 0$, then $H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*) = 1$.*

Lemma 3 gives us the following corollary (proved in Appendix C.5), which again will later help with our competitive ratio analysis of Algorithm 2.

COROLLARY 1. *Fix $k \in [K-1]$. Suppose $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ is the optimal solution of $\mathcal{P}_k^{\text{MCA}}$. For any advertiser $j \in V$, and $\tau \in [T]$, we have:*

- *(**Distribution Monotonicity**) $H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*) \ge \sum_{\tau' \ge \tau} \eta_{j,\tau}^{(k-1)}$.*
- *(**Only Disposing Smallest**) $\sum_{\tau' \ge \tau} \left(\eta_{j,\tau'}^{(k-1)} - y_{j,\tau'}^* \right) = \sum_{\tau' \ge \tau} \left(\eta_{j,\tau'}^{(k-1)} - y_{j,\tau'}^* \right) \cdot f_k \left(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*) \right)$.*

Properties of the Last Stage K. For stage K, note that $F_K(a) = 0$ for all $a \in [0,1]$ and thus convex program $\mathcal{P}_k^{\text{MCA}}$ degenerates to a linear program, which solves the optimal configuration allocation for the last stage graph under $\boldsymbol{\eta}^{(K-1)}$. This linear program $\mathcal{P}_k^{\text{MCA}}$ admits a dual as follows,

$$\min_{\boldsymbol{\lambda}, \boldsymbol{\theta}, \boldsymbol{\mu}, \boldsymbol{\pi} \ge 0} \sum_{i \in U_K} \lambda_i + \sum_{j \in V} \theta_j - \sum_{j \in V} \sum_{\tau \in [T]} \eta_{j,\tau}^{(K-1)} \cdot (\hat{w}_\tau - \pi_{j,\tau}) \text{ s.t.}$$
$$\mu_{i,c,j} + \theta_j \ge w_{i,c,j} \qquad i \in U_K, c \in C, j \in V \ , \qquad \text{(LP-}\mathcal{D}_K\text{)}$$
$$\theta_j + \pi_{j,\tau} \ge \hat{w}_\tau \qquad j \in V, \tau \in [T] \ ,$$
$$\lambda_i \ge \sum_{j \in V} \xi_{i,c,j} \cdot \mu_{i,c,j} \qquad i \in U_K, c \in C \ .$$

Technically speaking, the optimal solution of the dual program LP-$\mathcal{D}_K$ for the last stage satisfies the same KKT-like equalities/inequalities as the solution of the Lagrangian dual program $\mathcal{D}_k^{\text{MCA}}$ for the first $K-1$ stages. The main difference is the value of the optimal θ_j^*. As we showed in Lemma 2, $\theta_j^*=0$ in the first $K-1$ stages. This follows from the construction of function $F_k(\cdot)$, and in particular, the property that $\frac{d}{dx}F_k(1)=f_k(1)=1$. In contrast, $F_K\equiv 0$ in the last stage K, and the value of the optimal θ_j^* can be strictly positive. It turns out this makes the competitive ratio analysis of Algorithm 2 in the last stage a bit different from the previous stages. We introduce the following auxiliary lemma to characterize the required properties of dual program LP-$\mathcal{D}_K$ to help with this different part of the analysis. The proof is deferred to Appendix C.6.

LEMMA 4. *Suppose* $(\boldsymbol{\lambda}^*,\boldsymbol{\theta}^*,\boldsymbol{\mu}^*,\boldsymbol{\pi}^*)$ *is the optimal solution of the dual program LP-*$\mathcal{D}_K$*. For any advertiser* $j\in V$*,* $\tau\in[T]$*,*

- ***(Low Prices of Last Stage)*** *if* $\hat{w}_\tau\le\theta_j^*$*:* $\quad \eta_{j,\tau}^{(K-1)}\cdot\pi_{j,\tau}^*=0$
- ***(High Prices of Last Stage)*** *if* $\hat{w}_\tau>\theta_j^*$*:* $\quad \eta_{j,\tau}^{(K-1)}\cdot(\hat{w}_\tau-\pi_{j,\tau}^*)=\eta_{j,\tau}^{(K-1)}\cdot\theta_j^*$

4.3. Competitive Ratio Analysis of Algorithm 2

In this section we show the competitive ratio of $\Gamma(K)$ for Algorithm 2 as stated in Theorem 2. Remember the optimum offline LP PRIMAL-MCA in Section 2. We start by considering its dual:

$$\begin{aligned}
\max_{\mathbf{x},\mathbf{z}\ge\mathbf{0}}\quad & \sum_{i\in U}\sum_{c\in C}\sum_{j\in V} w_{i,c,j}\cdot x_{i,c,j} && \text{s.t.} \\
& \sum_{i\in U}\sum_{c\in C} x_{i,c,j}\le 1 && j\in V\ , \\
& \sum_{c\in C} z_{i,c}\le 1 && i\in U\ , \\
& x_{i,c,j}\le \xi_{i,c,j}\cdot z_{i,c} && i\in U, c\in C, j\in V\ ,
\end{aligned}
\qquad
\begin{aligned}
\min_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}\ge\mathbf{0}}\quad & \sum_{i\in U}\alpha_i+\sum_{j\in V}\beta_j && \text{s.t.} \\
& \gamma_{i,c,j}+\beta_j\ge w_{i,c,j} && i\in U, c\in C, j\in V\ , \\
& \alpha_i\ge\sum_{j\in V}\xi_{i,c,j}\cdot\gamma_{i,c,j} && i\in U, c\in C\ ,
\end{aligned}
\qquad \text{(OPT-PRIMAL-DUAL)}$$

Similar to Section 3.3, given the feasible primal assignment $\{\mathbf{x}^{(k)},\mathbf{y}^{(k)},\mathbf{z}^{(k)}\}_{k\in[K]}$ produced by Algorithm 2 (which leads to a final assignment of $\hat{\mathbf{x}}$ when considering the impressions disposals at the end), we construct a non-negative dual assignment so that,

- Primal and dual have the same objective values:

$$\sum_{i\in U}\sum_{c\in C}\sum_{j\in V} w_{i,c,j}\cdot\hat{x}_{i,c,j}=\sum_{k\in[K]}\left(\sum_{i\in U_k}\sum_{c\in C}\sum_{j\in V} w_{i,c,j}\cdot x_{i,c,j}^{(k)}-\sum_{j\in V}\sum_{\tau\in[T]}\hat{w}_\tau\cdot\left(\eta_{j,\tau}^{(k-1)}-y_{j,\tau}^{(k)}\right)\right)=\sum_{i\in U}\alpha_i+\sum_{j\in V}\beta_j$$

- Dual is approximately feasible:

$$\forall i\in U, c\in C, j\in V:\quad \gamma_{i,c,j}+\beta_j\ge\Gamma(K)\cdot w_{i,c,j}$$

and

$$\forall i\in U, c\in C:\quad \alpha_i\ge\sum_{j\in V}\xi_{i,c,j}\cdot\gamma_{i,c,j}$$

Existence of such a dual assignment will guarantee that Algorithm 2 is $\Gamma(K)$-competitive.

Our dual assignment is constructed stage-by-stage. First, set $\alpha_i \leftarrow 0$, $\beta_j \leftarrow 0$, and $\gamma_{i,c,j} \leftarrow 0$ for all $i \in U, c \in C, j \in V$. At each stage $k \in [K-1]$, let $(\boldsymbol{\lambda}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\mu}^{(k)}, \boldsymbol{\psi}^{(k)}, \boldsymbol{\phi}^{(k)})$ to denote $(\boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\mu}^*, \boldsymbol{\psi}^*, \boldsymbol{\phi}^*)$ in Lemma 2, and $H_{j,\tau}^{(k)}$ to denote $H_{j,\tau}(\mathbf{x}^{(k)}, \mathbf{y}^{(k)})$. Consider the following assignment corresponding to stages $k \in [K-1]$ for the dual program in OPT-PRIMAL-DUAL:

$$\begin{aligned}
&\forall i \in U_k: && \alpha_i \leftarrow \lambda_i^{(k)} \\
&\forall i \in U_k, c \in C, j \in V: && \gamma_{i,c,j} \leftarrow \mu_{i,c,j}^{(k)} \\
&\forall j \in V: && \beta_j \leftarrow \beta_j + \Delta\beta_j^{(k)} \\
& && \Delta\beta_j^{(k)} \triangleq \sum_{i \in U_k} \sum_{c \in C} x_{i,c,j}^{(k)} \left(\sum_{\tau \in [T]: \hat{w}(\tau) \le w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k\left(H_{j,\tau}^{(k)}\right) \right) \\
& && \qquad - \sum_{\tau \in [T]} \hat{w}_\tau \cdot \left(\eta_{j,\tau}^{(k-1)} - y_{j,\tau}^{(k)}\right)
\end{aligned}$$

For the last stage K, let $(\boldsymbol{\lambda}^{(K)}, \boldsymbol{\theta}^{(K)}, \boldsymbol{\mu}^{(K)}, \boldsymbol{\pi}^{(K)})$ be the optimal solution of LP-$\mathcal{D}_K$. We consider the following assignment corresponding to stage K for the dual program in OPT-PRIMAL-DUAL:

$$\begin{aligned}
&\forall i \in U_k: && \alpha_i \leftarrow \lambda_i^{(K)} \\
&\forall (i,c,j) \in E_k: && \gamma_{i,c,j} \leftarrow \mu_{i,c,j}^{(K)} \\
&\forall j \in V: && \beta_j \leftarrow \beta_j + \Delta\beta_j^{(K)} \\
& && \Delta\beta_j^{(K)} \triangleq \theta_j^{(K)} - \sum_{j \in V, \tau \in [T]} \eta_{j,\tau}^{(K-1)} \cdot (\hat{w}_\tau - \pi_{j,\tau}^{(K)})
\end{aligned}$$

The rest of the proof is quite technical and we postpone to Appendix C.7. Interestingly, the proof heavily relies on all the properties that we identified in Section 4.2; in particular, we use the *"Connection"* (Lemma 2) combined with the *"Disposing Smallest Price"* (Lemma 3) and its corollary (Corollary 1) to show for all stages $k \in [K-1]$ we have an equal change in the dual and primal objectives. For the last stage K, this property holds by construction. In order to show the approximate dual feasibility for constraints corresponding to stages $k \in [K-1]$, again we rely on the *"Connection"* (Lemma 2). However, showing dual approximate feasibility for the constraints corresponding to last stage is a bit different. There, we heavily use *"Low Prices of Last Stage"* (first part of Lemma 4) and *"High Prices of Last Stage"* (second part of Lemma 4) to do the analysis.

5. Competitive Ratio Upper-bound

In this section, we show a tight upper-bound of $\Gamma(K) = 1 - \left(1 - \frac{1}{K}\right)^K$ on the worst-case competitive ratio of any multi-stage fractional matching algorithm, even for the special case of unweighted matching. This upper-bound clearly applies to the case of integral algorithms and the case of vertex weighted matching problem, and shows our earlier lower-bounds are the best achievable competitive ratios.

THEOREM 3 **(Tight Upper-bound for Multi-stage Matching)**. *No multi-stage fractional matching algorithm can achieve a competitive ratio better than* $\Gamma(K) = 1 - \left(1 - \frac{1}{K}\right)^K$ *in the* K*-stage (unweighted) matching problem.*

High-level Proof Overview In order to show the upper-bound of $\Gamma(K)$, we propose a distribution over deterministic instances of the multi-stage weighted fractional matching problem (i.e., a randomized instance), and show no fractional multi-stage algorithm obtains a competitive ratio better than $\Gamma(K)$ over this randomized instance. To do so, we incorporate a simple factor revealing linear program whose optimal objective value provides an upper-bound on the competitive ratio of any feasible multi-stage matching algorithm. We then use LP primal-dual and by finding a feasible solution for its dual program we obtain the $\Gamma(K)$ upper-bound. Interestingly, our upper-bound instance is a distribution over unweighted instances. Hence, the same upper-bound holds for the special case of multi-stage fractional matching in which all weights are equal.

REMARK 5 (**Upper-bound for the Online Fractional Matching**). Notably, we need batches of size more than one in this instance, and hence our upper-bound *does not* apply to the online matching problem with K arrivals and batches of size 1. So, while the competitive ratio $\Gamma(K)$ is optimal for the multi-stage batch arrival problem, the competitive ratio of $\Gamma(n)$ might not be optimal for the special case of online matching with n vertex arrivals. To the best of our knowledge, the best known (almost) non-asymptotic competitive ratio upper-bound for the online fractional matching with n arrivals is due to Feige (2019), where a bound of $\Theta(n) = \left(1 - \frac{1}{e}\right)\left(1 + \frac{1}{2n}\right) + O\left(\frac{1}{n^2}\right)$ is established. Note that a simple asymptotic expansion shows that $\Gamma(n) = \left(1 - \frac{1}{e}\right) + \frac{1}{2en} + O\left(\frac{1}{n^2}\right) < \Theta(n)$, and hence the two bounds have a gap. Establishing the tight upper-bound for this problem is still open.

5.1. Proof of Theorem 3

Let $N = K^K$. Consider a bipartite graph $G = (U, V, E)$ with online vertices U and offline vertices V, where $U = V = \{1, \ldots, K \cdot N\} = [KN]$. Online vertices U are partitioned into K batches $\{U_1, \ldots, U_K\}$ where the size of U_k is $\left(\frac{K-1}{K}\right)^{k-1} N$ for $k \in [K-1]$, and $K\left(\frac{K-1}{K}\right)^{K-1} N$ for $k = K$. As a sanity check, notice that $\sum_{k=1}^{K-1}\left(\frac{K-1}{K}\right)^{k-1} N + K\left(\frac{K-1}{K}\right)^{K-1} N = KN$.

Let $\pi : [KN] \to [KN]$ be a uniform random permutation over $[KN]$. Define the edge set E of the graph G as follows: for each $k \in [K-1]$, each online vertex $i \in U_k$ and offline vertex $j \in V$, there is an edge (i, j) if $\pi(j) \le K\,|U_k|$; for each online vertex $i \in U_K$ and offline vertex $j \in V$, there is an edge (i, j) if $\pi(j) \le |U_K|$. Namely, at each stage k, we can partition offline vertices V into V_k and $\bar{V}_k$ where U_k and V_k form a complete bipartite subgraph while there is no edge between U_k and $\bar{V}_k$. The construction also ensures that $V_K \subseteq V_{K-1} \subseteq \cdots \subseteq V_1$. See Figure 4 for an illustration of this randomized problem instance when permutation $\pi(i) = NK - i + 1, i \in [NK]$ is realized.

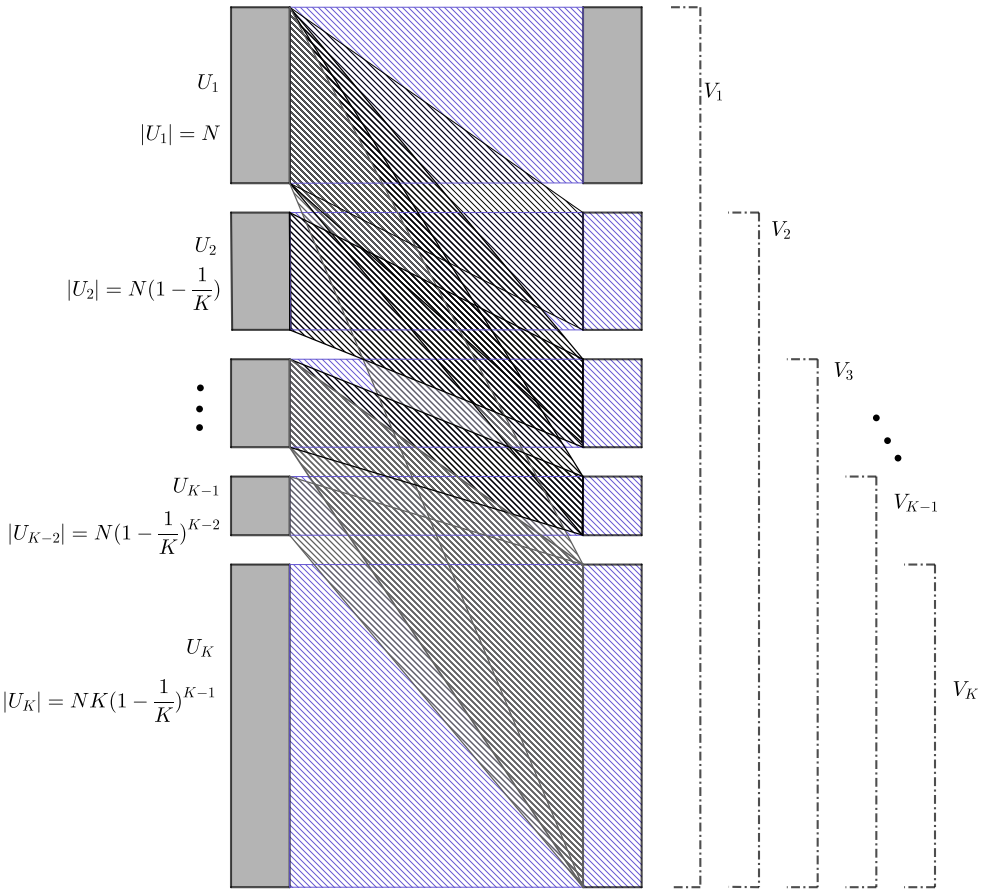


Figure 4 **The upper-bound instance when $\pi(i) = NK - i + 1, i \in [NK]$**

First, we claim that there exists a perfect matching: by backward induction on k starting from $k = K$, note that matching all online vertices in U_k with all offline vertices in $V_k \backslash V_{k+1}$ for $k = 1, 2, \ldots, K$ produces a perfect matching between U and V.

Next, we show that the expected size of the matching produced by any given (possibly randomized) multi-stage fractional matching algorithm is at most $\Gamma(K) \cdot KN$, where the expectation is taken over the randomness of the multi-stage algorithm and the permutation π. Let $\xi^{\pi}_{k,j}$ be the expected degree of offline vertex j matched by the algorithm at stage k under permutation π. Note that:

$$\mathbf{E}_{\pi}\left[\xi^{\pi}_{k,j} \mid j \notin V_k\right] = 0 \qquad \text{and} \qquad \mathbf{E}_{\pi}\left[\xi^{\pi}_{k,j} \mid j \in V_k\right] = \mathbf{E}_{\pi}\left[\xi^{\pi}_{k,j'} \mid j' \in V_k\right]$$

for all stages $k \in [K]$ and offline vertices $j, j' \in V$. The latter fact holds due to the symmetry of our graph construction, i.e., conditioned on the event $[j \in V_k \ \& \ j' \in V_k]$, it is an automorphism that exchanges j and j'. Thus, we can denote $\mathbf{E}_{\pi}\left[\xi^{\pi}_{k,j} \mid j \in V_k\right]$ by ξ_k, and write the following linear program and its dual to upper bound the expected size of the fractional matching produced by any given (randomized

or deterministic, integral or fractional) multi-stage algorithm:

$$\begin{array}{llll} \max & \sum_{k\in[K]} |V_k|\xi_k & \text{s.t.} & \\ & \sum_{k\in[K]} \xi_k \le 1 \ , & & \\ & |V_k|\xi_k \le |U_k| & k\in[K] \ , & \\ & \xi_k \ge 0 & k\in[K] \ . & \end{array} \qquad \begin{array}{llll} \min & \mu + \sum_{k\in[K]} |U_k|\zeta_k & \text{s.t.} & \\ & \mu + |V_k|\zeta_k \ge |V_k| & k\in[K] \ , & \\ & \mu \ge 0 \ , & & \\ & \zeta_k \ge 0 & k\in[K] \ . & \end{array}$$

Here, the variables are $\{\xi_k\}_{k\in[K]}$ in the primal program and $\{\mu, \zeta_k\}_{k\in[K]}$ in the dual program. The primal objective function is the expected size of fractional matching induced by $\{\xi_k\}_{k\in[K]}$, the first primal constraint ensures that the total expected allocation (over all K stages) is at most one for each offline node, and the second primal constraint ensures that the expected fractional allocation on the offline side is no more than the size of online nodes for each stage $k \in [K]$.

Consider the following feasible dual solution with objective value $\Gamma(K) \cdot KN$ in the dual program:

$$\mu = |V_K| \quad , \quad \zeta_k = \frac{|V_k| - |V_K|}{|V_k|}, \ \forall k \in [K].$$

Invoking the weak duality between the above primal-dual linear programs finishes the proof. □

6. Conclusion and Discussions

Motivated by the trade-off between batching and inefficiency in two-sided matching platforms, we studied the K-stage vertex weighted matching and its extension to the general model of configuration allocation with free-disposal. We designed optimal $(1-(1-1/K)^K)$-competitive fractional algorithms for these problems. A key ingredient of our algorithms and analysis is a sequence of polynomials of decreasing degrees, used as regularizers of maximum weight matching. We also presented the idea of price-level regularization to handle weight heterogeneity of the allocation on the offline side, and showed how our approach can incorporate this form of regularization. Our work also identifies deep connections between convex duality and primal-dual analysis of multi-stage resource allocation algorithms. There are several open questions and future research directions stemming from this work. See Appendix E for more discussions.

References

Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1253–1264. SIAM, 2011.

Shipra Agrawal and Nikhil R Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1405–1424. SIAM, 2014.

Shipra Agrawal, Morteza Zadimoghaddam, and Vahab Mirrokni. Proportional allocation: Simple, distributed, and diverse matching with high entropy. In *International Conference on Machine Learning*, pages 99–108. PMLR, 2018.

Saeed Alaei, Ali Makhdoumi, Azarakhsh Malekian, and Saša Pekeč. Revenue-sharing allocation strategies for two-sided media platforms: Pro-rata vs. user-centric. *Management Science*, 2022.

Susanne Albers and Achim Passen. New online algorithms for story scheduling in web advertising. *Algorithmica*, 81(1): 1–25, 2019.

Ali Aouad and Daniela Saban. Online assortment optimization for two-sided matching platforms. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 91–92, 2021.

Ali Aouad and Ömer Saritaç. Dynamic stochastic matching under limited time. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 789–790, 2020.

Nick Arnosti, Ramesh Johari, and Yash Kanoria. Managing congestion in matching markets. *Manufacturing & Service Operations Management*, 23(3):620–636, 2021.

Itai Ashlagi, Maximilien Burq, Chinmoy Dutta, Patrick Jaillet, Amin Saberi, and Chris Sholley. Edge weighted online windowed matching. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, pages 729–742, 2019.

Baris Ata, Nasser Barjesteh, and Sunil Kumar. Dynamic dispatch and centralized relocation of cars in ride-hailing platforms. *Available at SSRN 3675888*, 2020.

Angelos Aveklouris, Levi DeValve, Amy R Ward, and Xiaofan Wu. Matching impatient and heterogeneous demand and supply. *arXiv preprint arXiv:2102.02710*, 2021.

AWS. Amazon web services (aws) batch faqs, 2021. URL https://aws.amazon.com/batch/faqs/.

Lars Backstrom, Jon Kleinberg, and Ravi Kumar. Optimizing web traffic via the media scheduling problem. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, 2009.

Bahman Bahmani and Michael Kapralov. Improved bounds for online stochastic matching. In *European Symposium on Algorithms*, pages 170–181. Springer, 2010.

Santiago Balseiro, Haihao Lu, and Vahab Mirrokni. Regularized online allocation problems: Fairness and beyond. In *International Conference on Machine Learning*, pages 630–639. PMLR, 2021.

Santiago R Balseiro, Jon Feldman, Vahab Mirrokni, and Shan Muthukrishnan. Yield optimization of display advertising with ad exchange. *Management Science*, 60(12):2886–2907, 2014.

Santiago R Balseiro, Haihao Lu, and Vahab Mirrokni. The best of many worlds: Dual mirror descent for online allocation problems. *Operations Research*, 2022.

Dimitris Bertsimas, Dan A Iancu, and Pablo A Parrilo. Optimality of affine policies in multistage robust optimization. *Mathematics of Operations Research*, 35(2):363–394, 2010.

Dimitris Bertsimas, David B Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011.

John R Birge and Francois Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

Benjamin Birnbaum and Claire Mathieu. On-line bipartite matching made simple. *Acm Sigact News*, 39(1):80–87, 2008.

Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

Niv Buchbinder, Kamal Jain, and Joseph Seffi Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *European Symposium on Algorithms*, pages 253–264. Springer, 2007.

Moses Charikar, Chandra Chekuri, and Martin Pál. Sampling bounds for stochastic optimization. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 257–269. Springer, 2005.

Herman Chernoff et al. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, 23(4):493–507, 1952.

CNN. Amazon's new waste-reduction strategy: Deliver only once a week, 2021. URL https://www.cnn.com/2019/02/28/tech/amazon-day/index.html.

Anirban Dasgupta, Arpita Ghosh, Hamid Nazerzadeh, and Prabhakar Raghavan. Online story scheduling in web advertising. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1275–1284. SIAM, 2009.

Deloitte. Covid-19 and shifting generational preferences reshape the future of the us media and entertainment landscape, 2021. URL https://www2.deloitte.com/us/en/pages/about-deloitte/articles/press-releases/digital-media-trends.html.

Nikhil R Devanur and Thomas P Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *Proceedings of the 10th ACM conference on Electronic commerce*, pages 71–78, 2009.

Nikhil R Devanur and Kamal Jain. Online matching with concave returns. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 137–144, 2012.

Nikhil R Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 29–38, 2011.

Nikhil R Devanur, Kamal Jain, and Robert D Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 101–107. SIAM, 2013.

Nikhil R Devanur, Zhiyi Huang, Nitish Korula, Vahab S Mirrokni, and Qiqi Yan. Whole-page optimization and submodular welfare maximization with online bidders. *ACM Transactions on Economics and Computation (TEAC)*, 4(3):1–20, 2016.

Shaddin Dughmi, Jason Hartline, Robert D Kleinberg, and Rad Niazadeh. Bernoulli factories and black-box reductions in mechanism design. *Journal of the ACM (JACM)*, 68(2):1–30, 2021.

Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17:449–467, 1965.

Bruno Escoffier, Laurent Gourvès, Jérôme Monnot, and Olivier Spanjaard. Two-stage stochastic matching and spanning tree problems: Polynomial instances and approximation. *European Journal of Operational Research*, 205(1):19–30, 2010.

Hossein Esfandiari, Nitish Korula, and Vahab Mirrokni. Online allocation with traffic spikes: Mixing adversarial and stochastic models. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pages 169–186, 2015.

Uriel Feige. Tighter bounds for online bipartite matching. In *Building Bridges II*, pages 235–255. Springer, 2019.

Jon Feldman, Nitish Korula, Vahab Mirrokni, Shanmugavelayutham Muthukrishnan, and Martin Pál. Online ad assignment with free disposal. In *International workshop on internet and network economics*, pages 374–385. Springer, 2009a.

Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and Shan Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 117–126. IEEE, 2009b.

Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S Mirrokni, and Cliff Stein. Online stochastic packing applied to display ad allocation. In *European Symposium on Algorithms*, pages 182–194. Springer, 2010.

Yiding Feng, Rad Niazadeh, and Amin Saberi. Linear programming based online policies for real-time assortment of reusable resources. *Available at SSRN 3421227*, 2019.

Yiding Feng, Rad Niazadeh, and Amin Saberi. Near-optimal bayesian online assortment of reusable resources. *Chicago Booth Research Paper*, (20-40), 2020a.

Yiding Feng, Rad Niazadeh, and Amin Saberi. Two-stage matching and pricing with applications to ride hailing. *Available at SSRN*, 2020b.

Yiding Feng, Rad Niazadeh, and Amin Saberi. Online assortment of reusable resources with exogenous replenishment. *Available at SSRN 3795056*, 2021.

Yiding Feng, Rad Niazadeh, and Amin Saberi. Near-optimal bayesian online assortment of reusable resources. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 964–965, 2022.

João Lucas Hana Frade, Jorge Henrique Caldeira de Oliveira, and Janaina de Moura Engracia Giraldi. Advertising in streaming video: An integrative literature review and research agenda. *Telecommunications Policy*, 45(9):102186, 2021.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2): 95–110, 1956.

Marguerite Frank, Philip Wolfe, et al. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3 (1-2):95–110, 1956.

Tibor Gallai. Maximale systeme unabhangiger kanten. *Magyar Tud. Akad. Mat. Kutato Int. Kozl.*, 9:401–413, 1964.

Ashish Goel, Michael Kapralov, and Sanjeev Khanna. On the communication and streaming complexity of maximum bipartite matching. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 468–485. Society for Industrial and Applied Mathematics, 2012.

Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In *SODA*, volume 8, pages 982–991. Citeseer, 2008.

Negin Golrezaei and Evan Yao. Online resource allocation with time-flexible customers. *arXiv preprint arXiv:2108.03517*, 2021.

Negin Golrezaei, Hamid Nazerzadeh, and Paat Rusmevichientong. Real-time optimization of personalized assortments. *Management Science*, 60(6):1532–1551, 2014.

Xiao-Yue Gong, Vineet Goyal, Garud N Iyengar, David Simchi-Levi, Rajan Udwani, and Shuangyu Wang. Online assortment optimization with reusable resources. *Management Science*, 68(7):4772–4785, 2022.

Vineet Goyal and Rajan Udwani. Online matching with stochastic rewards: Optimal competitive ratio via path based formulation. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 791–791, 2020.

Vineet Goyal, Garud Iyengar, and Rajan Udwani. Online allocation of reusable resources via algorithms guided by fluid approximations. *arXiv preprint arXiv:2010.03983*, 2020.

Grani A Hanasusanto and Daniel Kuhn. Conic programming reformulations of two-stage distributionally robust linear programs over wasserstein balls. *Operations Research*, 66(3):849–869, 2018.

Grani A Hanasusanto, Daniel Kuhn, and Wolfram Wiesemann. K-adaptability in two-stage robust binary programming. *Operations Research*, 63(4):877–891, 2015.

Omar El Housni, Vineet Goyal, Oussama Hanguir, and Clifford Stein. Matching drivers to riders: A two-stage robust approach. *arXiv preprint arXiv:2011.03624*, 2020.

Zhiyi Huang and Qiankun Zhang. Online primal dual meets online matching with stochastic rewards: configuration lp to the rescue. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1153–1164, 2020.

Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, and Xue Zhu. How to match when all vertices arrive online. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 17–29, 2018.

Nicole Immorlica, David Karger, Maria Minkoff, and Vahab S Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 691–700. Society for Industrial and Applied Mathematics, 2004.

Patrick Jaillet and Xin Lu. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research*, 39(3):624–646, 2014.

Manas Joshi, Arshdeep Singh, Sayan Ranu, Amitabha Bagchi, Priyank Karia, and Puneet Kala. Batching and matching for food delivery in dynamic road networks. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 2099–2104. IEEE, 2021.

Bala Kalyanasundaram and Kirk R Pruhs. An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science*, 233(1-2):319–325, 2000.

Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 352–358, 1990.

Irit Katriel, Claire Kenyon-Mathieu, and Eli Upfal. Commitment under uncertainty: Two-stage stochastic matching problems. *Theoretical Computer Science*, 408(2-3):213–223, 2008.

Nan Kong and Andrew J Schaefer. A factor 12 approximation algorithm for two-stage stochastic matching problems. *European Journal of Operational Research*, 172(3):740–746, 2006.

Daniel Kuhn. *Generalized bounds for convex multistage stochastic programs*, volume 548. Springer Science & Business Media, 2006.

Euiwoong Lee and Sahil Singla. Maximum matching in the online batch-arrival model. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 355–367. Springer, 2017.

Tie-Yan Liu, Weidong Ma, Tao Qin, Pingzhong Tang, Guang Yang, and Bo Zheng. Online non-preemptive story scheduling in web advertising. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 269–277, 2016.

Lyft. Matchmaking in Lyft line: Part 1, 2016. URL https://eng.lyft.com/matchmaking-in-lyft-line-9c2635fe62c4.

Will Ma and David Simchi-Levi. Algorithms for online matching, assortment, and pricing with tight weight-dependent competitive ratios. *Operations Research*, 2020.

Will Ma, David Simchi-Levi, and Jinglong Zhao. Dynamic pricing (and assortment) under a static calendar. *Management Science*, 67(4):2292–2313, 2021.

Vahideh Manshadi and Scott Rodilitz. Online policies for efficient volunteer crowdsourcing. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 315–316, 2020.

Vahideh H Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research*, 37(4):559–573, 2012.

Aranyak Mehta. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4): 265–368, 2013.

Aranyak Mehta and Debmalya Panigrahi. Online matching with stochastic rewards. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 728–737. IEEE, 2012.

Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. *Journal of the ACM (JACM)*, 54(5):22–es, 2007.

Aranyak Mehta, Bo Waggoner, and Morteza Zadimoghaddam. Online stochastic matching with unequal probabilities. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1388–1404. SIAM, 2014.

Erhun Özkan and Amy R Ward. Dynamic matching for real-time ride sharing. *Stochastic Systems*, 10(1):29–70, 2020.

Paat Rusmevichientong, Mika Sumida, and Huseyin Topaloglu. Dynamic assortment optimization for reusable products with random usage durations. *Management Science*, 66(7):2820–2844, 2020.

Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

David B Shmoys and Mauro Sozio. Approximation algorithms for 2-stage stochastic scheduling problems. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 145–157. Springer, 2007.

David B Shmoys and Chaitanya Swamy. Stochastic optimization is (almost) as easy as deterministic optimization. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 228–237. IEEE, 2004.

Hanna Sumita, Yasushi Kawase, Sumio Fujita, Takuro Fukunaga, and RA Center. Online optimization of video-ad allocation. In *IJCAI*, pages 423–429, 2017.

Uber. Uber marketplace and matching, 2020. URL https://marketplace.uber.com/matching.

Alberto Vera and Siddhartha Banerjee. The bayesian prophet: A low-regret framework for online decision making. *Management Science*, 67(3):1368–1391, 2021.

Yaqi Xie, Will Ma, and Linwei Xin. The benefits of delay to online decision-making. *Available at SSRN*, 2022.

Google YouTube. Personal communication with the YouTube advertising product teams., 2021.

Lingyu Zhang, Tao Hu, Yue Min, Guobin Wu, Junying Zhang, Pengcheng Feng, Pinghua Gong, and Jieping Ye. A taxi order dispatch model based on combinatorial optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 2151–2159, 2017.

Appendix

A. Connection Between Baseline and Extension Models

We formally show that the multi-stage configuration allocation generalizes the multi-stage vertex weighted matching. To see this, we consider a reduction in which every instance in the vertex weighted matching can be thought as an instance in the configuration allocation as follows: Each advertiser j has a unit demand ($n_j \leftarrow 1$). The feasible configuration set encodes all edges E in the baseline model ($C \leftarrow E$). Moreover, for every edge $(i^\dagger, j^\dagger) \in E$, the corresponding configuration c encodes the number of reserved impression $\{\xi_{i,c,j}\}_{i,j}$ as $\xi_{i,c,j} \leftarrow \mathbb{1}\{i = i^\dagger, j = j^\dagger\}$. The per-impression prices are identical for each advertiser j and only depend on the weight w_j of the offline vertex j ($w_{i,c,j} \leftarrow w_j \mathbb{1}\{(i,j) \in E\}$). Finally, note that since per-impression price are identical for each advertiser and number of reserved impression $\{\xi_{i,c,j}\}$ is encoded as the aforementioned indicator function, without loss of generality, we can restrict attention to online algorithms whose allocations never exceed demand $\{n_j\}$ of advertisers, and thus no preemption or free disposal is needed.

B. Results in Other Bipartite Allocation Models

In this section, we revisit three classic bipartite allocation models: b-matching problem (Kalyanasundaram and Pruhs, 2000), AdWords problem (Mehta et al., 2007), and assortment problem (Golrezaei et al., 2014). We discuss how our results extend in these models.

B.1. B-Matching Problem

In this subsection, we consider the K-stage vertex weighted b-matching problem.

Model Description The multi-stage vertex weighted b-matching problem is a variant of the multi-stage vertex weighted matching where the matching constraint of each offline vertex $j \in V$ is replaced with a capacity constraint B_j. Let $B_{\min} = \min_{j \in V} B_j$ be the smallest capacity.

Connection to Configuration Allocation The B-matching problem is a special case of the configuration allocation. To see this, we consider a reduction (similar to the one in Section A) in which every instance in the vertex weighted b-matching can be thought as an instance in the configuration allocation as follows: Each advertiser j has a unit demand ($n_j \leftarrow B_j$). The feasible configuration set encodes all edges E in the b-matching problem ($C \leftarrow E$). Moreover, for every edge $(i^\dagger, j^\dagger) \in E$, the corresponding configuration c encodes the number of reserved impression $\{\xi_{i,c,j}\}_{i,j}$ as $\xi_{i,c,j} \leftarrow \mathbb{1}\{i = i^\dagger, j = j^\dagger\}$. The per-impression prices are identical for each advertiser j and only depend on the weight w_j of the offline vertex j ($w_{i,c,j} \leftarrow w_j \mathbb{1}\{(i,j) \in E\}$). Finally, note that since per-impression price are identical for each advertiser and number of reserved impression $\{\xi_{i,c,j}\}$ is encoded as the aforementioned indicator function, without loss of generality, we can restrict attention to online algorithms whose allocations never exceed demand $\{n_j\}$ of advertisers, and thus no preemption or free disposal is needed.

Fractional Multi-stage B-matching Given the connection to the configuration allocation problem, we can implement Algorithm 2 where the convex program $\mathcal{P}_k^{\text{MCA}}$ can be simplified as program $\mathcal{P}_k^{\text{VWM}}$ defined below,

$$\begin{aligned}
\mathbf{x}^{(k)} = \arg\max_{\mathbf{x}} \quad & \sum_{(i,j)\in E_k} w_j \frac{x_{ij}}{B_j} - \sum_{j\in V} w_j F_k\left(\frac{y_j}{B_j} + \sum_{i:(i,j)\in E_k} \frac{x_{ij}}{B_j}\right) \text{ s.t.} \\
& \sum_{j:(i,j)\in E_k} x_{ij} \le 1 && i \in U_k \\
& \sum_{i:(i,j)\in E_k} x_{ij} \le B_j - y_j && j \in V \\
& x_{ij} \ge 0 && (i,j) \in E_k
\end{aligned} \tag{$\mathcal{P}_k^{\text{MWBM}}$}$$

Since the feasible configuration sets encodes all edges E in the b-matching problem, variables $\mathbf{z}$ in program $\mathcal{P}_k^{\text{MCA}}$ become redundant and removed in program $\mathcal{P}_k^{\text{VWM}}$. Moreover, since there is no need for preemption in the b-matching problem, variables $\mathbf{y}$ in program $\mathcal{P}_k^{\text{MCA}}$ become redundant and we reuse this notation to denote the allocation from previous stages in program $\mathcal{P}_k^{\text{VWM}}$.

Similar to Section 3, the convex program $\mathcal{P}_k^{\text{MWBM}}$ introduces the following structural decomposition (Lemma 5). We omits the proof of this lemma, as it follows exactly the same argument as its analog in the vertex weighted matching problem (Proposition 2). The main difference between Lemma 5 and Proposition 2 is the closed-form expression in uniformity property.

LEMMA 5 **(Structural Decomposition/Vertex Weighted b-matching)**. *Fix any $k \in [K-1]$. Let $\mathbf{x}^{(k)}$ be the optimal solution of the convex program $\mathcal{P}_k^{MWBM}$. Consider the subgraph $G'_k = (U_k, V, E'_k)$ of G_k, where $E'_k = \{(i,j) \in E_k : x_{ij}^{(k)} > 0\}$. Let $V_{k,0}$ be the set of offline vertices in V who exhausts their capacities by $\mathbf{x}^{(k)}$ given the current $\mathbf{y}$, and $U_{k,0}$ be the set of online vertices who are neighbors of $V_{k,0}$ in G'_k, that is,*

$$V_{k,0} \triangleq \{j \in V : y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)} = B_j\} \quad \text{and} \quad U_{k,0} \triangleq \{i \in U_k : \exists j \in V_{k,0}, x_{ij}^{(k)} > 0\} \ .$$

Moreover, let the pairs $\{(U_{k,\ell}, V_{k,\ell})\}_{\ell=1}^{L_k}$ identify the L_k connected components of the induced subgraph $G'_k[U_k \setminus U_{k,0}, V \setminus V_{k,0}]$ of G'_k. Then:

1. *Uniformity: $\forall\ \ell \in [0 : L_k], j, j' \in V_{k,\ell}$:*
$$w_j\left(1 - f_k\left(\frac{y_j}{B_j} + \sum_{i:(i,j)\in E_k} \frac{x_{ij}^{(k)}}{B_j}\right)\right) = w_{j'}\left(1 - f_k\left(\frac{y_{j'}}{B_{j'}} + \sum_{i:(i,j')\in E_k} \frac{x_{ij'}^{(k)}}{B_{j'}}\right)\right) \triangleq c_\ell^{(k)}$$
2. *Monotonicity: $\forall\ \ell, \ell' \in [0 : L_k]$: there exists no edge in E_k between $U_{k,\ell}$ and $V_{k,\ell'}$ if $c_\ell^{(k)} < c_{\ell'}^{(k)}$.*
3. *Saturation: $\forall\ \ell \in [L_k]$: all online vertices in $U_{k,\ell}$ are fully matched by $\mathbf{x}^{(k)}$, i.e., $\sum_{j:(i,j)\in E_k} x_{ij}^{(k)} = 1, i \in U_{k,\ell}$.*

Rounding to an Integral Allocation To output integral solutions in the K-stage vertex weighted b-matching problem, we further round the fractional solution $\mathbf{x}^{(k)}$ to an integral solution $\hat{\mathbf{x}}^{(k)}$ for each stage k as follows. Let $\{(U_{k,0}, V_{k,0}), \ldots, (U_{k,L_k}, V_{k,L_k})\}$ be the structural decomposition of (U_k, V, E_k) in Lemma 5. For any edge $(i,j) \in E_k$ such that $i \in U_{k,\ell}$, $j \in V_{k,\ell'}$ and $\ell \neq \ell'$, we set $\hat{x}_{ij}^{(k)} \leftarrow x_{ij}^{(k)} = 0$. Denote the edges between $U_{k,\ell}$ and $V_{k,\ell}$ by $E_{k,\ell}$. For each subgraph $(U_{k,\ell}, V_{k,\ell}, E_{k,\ell})$, we set $\{\hat{x}_{ij}^{(k)}\}_{(i,j)\in E_{k,\ell}}$ to be an *optimal vertex* of the following linear

program $\mathcal{P}^{\text{rounding}}_{k,\ell}$ (which can be found in polynomial time, for example by the simplex algorithm or ellipsoid method):

$$\begin{aligned}
\{\hat{x}^{(k)}_{ij}\}_{(i,j)\in E_{k,\ell}} = \arg\max_{\mathbf{x}} \quad & \sum_{(i,j)\in E_{k,\ell}} x_{ij} && \text{s.t.} \\
& \sum_{j:(i,j)\in E_{k,\ell}} x_{ij} \le 1 && i \in U_{k,\ell} \\
& \sum_{i:(i,j)\in E_{k,\ell}} x_{ij} \ge \left\lfloor \sum_{i:(i,j)\in E_{k,\ell}} x^{(k)}_{ij} \right\rfloor && j \in V_{k,\ell} \qquad (\mathcal{P}^{\text{rounding}}_{k,\ell}) \\
& \sum_{i:(i,j)\in E_{k,\ell}} x_{ij} \le \left\lfloor \sum_{i:(i,j)\in E_{k,\ell}} x^{(k)}_{ij} \right\rfloor + 1 && j \in V_{k,\ell} \\
& x_{ij} \ge 0 && (i,j) \in E_{k,\ell}
\end{aligned}$$

where $\lfloor a \rfloor$ is the floor function that outputs the greatest integer less than or equal to a. Note that $\mathcal{P}^{\text{rounding}}_{k,\ell}$ is a feasible program since $\{x^{(k)}_{ij}\}_{(i,j)\in E_{k,\ell}}$ is a feasible solution. Moreover, it can be easily checked that the matrix defining the feasibility polytope in $\mathcal{P}^{\text{rounding}}_{k,\ell}$, with entries in $\{0, \pm 1\}$, is indeed a Totally Unimodular Matrix (TUM) (cf. Schrijver, 2003). As a result, because all the lower-bound and upper-bound constraints are integers, all the vertices of this feasibility polytope are going to be integral. Thus, $\hat{\mathbf{x}}^{(k)}$ is an integral optimal solution.

We are now ready to present the competitive ratio result of the multi-stage vertex weighted b-matching problem. We propose both fractional and integral versions of the *Polynomial Regularized Maximum Weight B-matching Algorithm* (`PR-MWBM`) – see Algorithm 3 for a formal description.

Proposition 3 **(Optimal Multi-stage Algorithm for Weighted Bipartite B-matching)**. *In the K-stage fractional (resp. integral) vertex weighted b-matching problem, the fractional (resp. integral) version of `PR-MWBM` (Algorithm 3) is $\Gamma(K)$-competitive (resp. $(\Gamma(K) - O(1/B_{min}))$-competitive), where $\Gamma(K) = 1 - \left(1 - \frac{1}{K}\right)^K$.*

The competitive ratio of the fractional solution in Proposition 3 is directly implied by Theorem 2 since the fractional b-matching problem is a special case of the fractional configuration allocation problem. The proof of the competitive ratio of the integral solution in Proposition 3 is similar to the one for Theorem 1, and we include it below for completeness.

Proof of Proposition 3 for integral solutions. For each stage $k \in [K-1]$, let $\{(U_{k,0}, V_{k,0}), \ldots, (U_{k,L_k}, V_{k,L_k})\}$ be the structural decomposition of (U_k, V, E_k) as in Lemma 5, and $\{x^{(k)}_{ij}\}_{(i,j)\in E_k}$ be the optimal (fractional) solution of the program $\mathcal{P}^{\text{MWBM}}_k$ and $\{\hat{x}^{(k)}_{ij}\}_{(i,j)\in E_k}$ be the optimal (integral) solution of the program $\mathcal{P}^{\text{rounding}}_{k,\ell}$. First, we define the following notations which will be useful in this proof:

$$\forall j \in V,\ k \in [K]: \qquad \hat{y}^{(k)}_j \triangleq \sum_{s=1}^{k-1} \sum_{i:(i,j)\in E_s} \hat{x}^{(s)}_{ij}, \qquad y^{(k)}_j \triangleq \sum_{s=1}^{k-2} \sum_{i:(i,j)\in E_s} \hat{x}^{(s)}_{ij} + \sum_{i:(i,j)\in E_{k-1}} x^{(k-1)}_{ij}$$

Now consider the offline primal-dual linear program Primal-Dual-bmatching.

$$\begin{aligned}
\max \quad & \sum_{(i,j)\in E} w_j x_{ij} && \text{s.t.} \\
& \sum_{j:(i,j)\in E} x_{ij} \le 1 && i \in U\ , \\
& \sum_{i:(i,j)\in E} x_{ij} \le B_j && j \in V\ , \\
& x_{ij} \ge 0 && (i,j) \in E\ .
\end{aligned}
\qquad
\begin{aligned}
\min \quad & \sum_{i\in U} \alpha_i + \sum_{j\in V} B_j \beta_j && \text{s.t.} \\
& \alpha_i + \beta_j \ge w_j && (i,j) \in E\ , \\
& \alpha_i \ge 0 && i \in U\ , \\
& \beta_j \ge 0 && j \in V\ .
\end{aligned}
\qquad (\text{Primal-Dual-bmatching})$$

Algorithm 3: Polynomial Regularized Max Weight b-Matching (`PR-MWBM`)

Input: Number of batches/stages K, Integral solution flag *`Flag`*.

$\forall j \in V:\ y_j \leftarrow 0$.

for *stage* $k=1$ *to* K **do**

  `/*` k^{th} `batch of vertices in` U`, denoted by` U_k`, arrives. Let` E_k `denote the set of edges incident to vertices in` U_k`. */`

  Given $\{y_j\}_{j\in V}$ and subgraph $G_k=(U_k,V,E_k)$, solve $\mathbf{x}^{(k)}$ from the convex program $\mathcal{P}_k^{\text{MWBM}}$.

  if *`Flag`* $==$ *False* **then**

    $\forall j\in V: y_j \leftarrow y_j + \sum_{i\in U_k} x_{ij}^{(k)}$.

    Return $\mathbf{x}^{(k)}$ as the fractional b-matching allocation of stage k.

  else

    Given $\mathbf{x}^{(k)}$, and structural decomposition $\{(U_{k,\ell},V_{k,\ell}\}_{\ell\in[0:L_k]}$, solve optimal vertex $\hat{\mathbf{x}}^{(k)}$ from the linear program $\mathcal{P}_{k,\ell}^{\text{rounding}}$ for all $\ell\in[0:L_k]$.

    $\forall j\in V: y_j \leftarrow y_j + \sum_{i\in U_k} \hat{x}_{ij}^{(k)}$.

    Return $\hat{\mathbf{x}}^{(k)}$ as the integral b-matching allocation of stage k.

Once again, given the feasible primal (integral) assignment $\{\hat{\mathbf{x}}^{(k)}\}_{k\in[K]}$ produced by Algorithm 3, we construct a dual assignment so that,

- Primal and dual have the same objective values: $\sum_{k\in[K]}\sum_{(i,j)\in E_k} w_j \hat{x}_{ij}^{(k)} = \sum_{i\in U}\alpha_i + \sum_{j\in V} B_j\beta_j$,
- Dual is approximately feasible: $\forall (i,j)\in E: \alpha_i+\beta_j \geq (\Gamma(K)-O(1/B_{\min}))\cdot w_j$.

[*Step i*] *Dual assignment construction.* First, set $\alpha_i\leftarrow 0$ and $\beta_j\leftarrow 0$ for all $i\in U, j\in V$. At each stage $k\in[0:K-1]$, update the dual variables as follows:

$$\forall i\in U_{k,\ell},\ \ell\in[0:L_k]: \qquad \alpha_i \leftarrow \frac{1}{|U_{k,\ell}|}\sum_{j\in V_{k,\ell}}\left(\hat{y}_j^{(k+1)}-\hat{y}_j^{(k)}\right)\left(1-f_k\left(\frac{\hat{y}_j^{(k+1)}-1}{B_j}\right)\right)w_j$$

$$\forall j\in V_{k,\ell},\ \ell\in[0:L_k]: \qquad \beta_j \leftarrow \beta_j + \Delta\beta_j^{(k)}$$

$$\Delta\beta_j^{(k)} \triangleq \left(\frac{\hat{y}_j^{(k+1)}-\hat{y}_j^{(k)}}{B_j}\right) f_k\left(\frac{\hat{y}_j^{(k+1)}-1}{B_j}\right) w_j$$

For the last stage K, note that $F_K(x)=0$ for all $x\in[0,1]$. Therefore, Algorithm 3 essentially solves the maximum vertex weighted b-matching in the last stage graph (U_K,V,E_K) where each $j\in V$ has initial capacity $B_j-\hat{y}_j^{(K)}$. Let $\{\hat{\alpha}_i^{(K)},\hat{\beta}_j^{(K)}\}_{i\in U_K,j\in V}$ be the *optimal* dual solution of this LP. Then update the dual variables as follows:

$$\forall i\in U_K: \quad \alpha_i \leftarrow \hat{\alpha}_i^{(K)}$$

$$\forall j\in V \ : \quad \beta_j\leftarrow\beta_j+\Delta\beta_j^{(K)},\quad \Delta\beta_j^{(K)} \triangleq \left(1-\frac{\hat{y}_j^{(K)}}{B_j}\right)\hat{\beta}_j^{(K)}$$

[*Step ii*] *Comparing objective values in primal and dual.* Note that by construction, the objective values in primal and dual are equal (for each stage).

[*Step iii*] *Checking approximate feasibility of dual.* To show approximate dual feasibility, it is enough to show that for any stage $k \in [K]$ and any arbitrary edge $(i,j) \in E_k$,

$$\alpha_i + \sum_{s=1}^{k} \Delta\beta_j^{(s)} \geq w_j \left(1 - (1-1/K)^K - 3/B_{\min}\right) \tag{2}$$

We next show inequality (2) holds for $k \in [K-1]$ and $k = K$ with slightly different arguments. For any $k \in [K-1]$, $\ell \in [0 : L_k]$, fix any edge $(i^\dagger, j^\dagger) \in E_k$ with online vertex $i^\dagger \in U_{k,\ell}$. We next show the following three inequalities (3), (4) and (5) hold.

$$\begin{aligned} &\frac{1}{|U_{k,\ell}|} \sum_{j \in V_{k,\ell}} \left(y_j^{(k+1)} - \hat{y}_j^{(k)}\right) \left(1 - f_k\left(\frac{y_j^{(k+1)}}{B_j}\right)\right) w_j \\ &\quad + \sum_{s=1}^{k-1} \left(\frac{\hat{y}_{j^\dagger}^{(s+1)} - \hat{y}_{j^\dagger}^{(s)}}{B_{j^\dagger}}\right) \cdot f_s\left(\frac{\hat{y}_{j^\dagger}^{(s+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} + \left(\frac{y_{j^\dagger}^{(k+1)} - \hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}}\right) \cdot f_k\left(\frac{y_{j^\dagger}^{(k+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} \\ &\quad \geq \left(1 - (1-1/K)^K\right) w_{j^\dagger} \end{aligned} \tag{3}$$

and

$$\alpha_{i^\dagger} \geq \frac{1}{|U_{k,\ell}|} \sum_{j \in V_{k,\ell}} \left(y_j^{(k+1)} - \hat{y}_j^{(k)}\right) \left(1 - f_k\left(\frac{y_j^{(k+1)}}{B_j}\right)\right) w_j \tag{4}$$

and

$$\begin{aligned} &\sum_{s=1}^{k} \Delta\beta_{j^\dagger}^{(s)} + \frac{3w_{j^\dagger}}{B_{j^\dagger}} \\ &\geq \sum_{s=1}^{k-1} \left(\frac{\hat{y}_{j^\dagger}^{(s+1)} - \hat{y}_{j^\dagger}^{(s)}}{B_{j^\dagger}}\right) \cdot f_s\left(\frac{\hat{y}_{j^\dagger}^{(s+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} + \left(\frac{y_{j^\dagger}^{(k+1)} - \hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}}\right) \cdot f_k\left(\frac{y_{j^\dagger}^{(k+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} \end{aligned} \tag{5}$$

Note that combining inequalities (3), (4) and (5) finishes the argument for inequality (2) on edge $(i^\dagger, j^\dagger) \in E_k$ with $k \in [K-1]$.

Proof of inequality (3). Let $\ell' \in [0 : L_k]$ be the index s.t. offline vertex $j^\dagger \in V_{k,\ell'}$. Due to monotonicity property in Lemma 5, we have $\ell \geq \ell'$ and $c_\ell^{(k)} \geq c_{\ell'}^{(k)}$. First, note that

$$c_\ell^{(k)} = \frac{1}{|U_{k,\ell}|} \sum_{j \in V_{k,\ell}} \left(y_j^{(k+1)} - \hat{y}_j^{(k)}\right) \left(1 - f_k\left(\frac{y_j^{(k+1)}}{B_j}\right)\right) w_j$$

due to uniformity and saturation in Lemma 5.[18] Hence, we can rewrite inequality (3) as

$$\begin{aligned} &c_\ell^{(k)} + \sum_{s=1}^{k-1} \left(\frac{\hat{y}_{j^\dagger}^{(s+1)} - \hat{y}_{j^\dagger}^{(s)}}{B_{j^\dagger}}\right) \cdot f_s\left(\frac{\hat{y}_{j^\dagger}^{(s+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} + \left(\frac{y_{j^\dagger}^{(k+1)} - \hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}}\right) \cdot f_k\left(\frac{y_{j^\dagger}^{(k+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} \\ &\quad \geq \left(1 - (1-1/K)^K\right) w_{j^\dagger} \end{aligned}$$

which holds since $c_\ell^{(k)} \geq c_{\ell'}^{(k)}$ and Lemma 1 for the sequence

$$0 = \hat{y}_{j^\dagger}^{(1)}/B_{j^\dagger} \leq \hat{y}_{j^\dagger}^{(2)}/B_{j^\dagger} \leq \ldots \leq \hat{y}_{j^\dagger}^{(k)}/B_{j^\dagger} \leq 1 - \sum_{i:(i,j^\dagger) \in E_k} x_{ij^\dagger}^{(k)}/B_{j^\dagger} \ .$$

[18] For $\ell = 0$, the construction of $(U_{k,0}, V_{k,0})$ suggests that $y_j^{(k+1)} = B_j$ for all $j \in V_{k,0}$, and thus $f_k\left(\frac{y_j^{(k+1)}}{B_j}\right) = 1$, which implies that the right hand side equals $c_k^{(0)} = 0$. For $l \in [L_k]$, since $\sum_{j \in V_{k,\ell}} \left(y_j^{(k+1)} - \hat{y}_j^{(k)}\right) = |U_{k,\ell}|$ (due to saturation) and for all $j \in V_{k,\ell}$ $w_j \left(1 - f_k\left(\frac{y_j^{(k+1)} - \hat{y}_j^{(k)}}{B_j}\right)\right) = c_\ell^{(k)}$ (due to uniformity), the right hand side equals $c_\ell^{(k)}$.

Proof of inequality (4). Since $f_k(\cdot)$ is weakly increasing, and $y_j^{(k+1)} \geq \hat{y}_j^{(k+1)} - 1$ by construction, we have $f_k\left(\frac{y_j^{(k+1)}}{B_j}\right) \geq f_k\left(\frac{\hat{y}_j^{(k+1)}-1}{B_j}\right)$. Thus,

$$\begin{aligned}
\alpha_{i^\dagger} &= \frac{1}{|U_{k,\ell}|} \sum_{j \in V_{k,\ell}} \left(\hat{y}_j^{(k+1)} - \hat{y}_j^{(k)}\right) \left(1 - f_j\left(\frac{\hat{y}_j^{(k+1)} - 1}{B_j}\right)\right) w_j \\
&\geq \frac{1}{|U_{k,\ell}|} \sum_{j \in V_{k,\ell}} \left(\hat{y}_j^{(k+1)} - \hat{y}_j^{(k)}\right) \left(1 - f_j\left(\frac{y_j^{(k+1)}}{B_j}\right)\right) w_j \\
&= \frac{1}{|U_{k,\ell}|} \sum_{j \in V_{k,\ell}} \left(y_j^{(k+1)} - \hat{y}_j^{(k)}\right) \left(1 - f_k\left(\frac{y_j^{(k+1)}}{B_j}\right)\right) w_j
\end{aligned}$$

where the last equality uses the fact that $\sum_{j \in V_{k,\ell}} \left(\hat{y}_j^{(k+1)} - \hat{y}_j^{(k)}\right) = \sum_{j \in V_{k,\ell}} \left(y_j^{(k+1)} - \hat{y}_j^{(k)}\right)$ (by the construction of $\hat{\mathbf{x}}^*$) and uniformity in Lemma 5.

Proof of inequality (5). We start with the right hand side of inequality (5),

$$\begin{aligned}
&\sum_{s=1}^{k-1} \left(\frac{\hat{y}_{j^\dagger}^{(s+1)} - \hat{y}_{j^\dagger}^{(s)}}{B_{j^\dagger}}\right) \cdot f_s\left(\frac{\hat{y}_{j^\dagger}^{(s+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} + \left(\frac{y_{j^\dagger}^{(k+1)} - \hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}}\right) \cdot f_k\left(\frac{y_{j^\dagger}^{(k+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} \\
\leq &\sum_{s=1}^{k-1} \left(\frac{\hat{y}_{j^\dagger}^{(s+1)} - \hat{y}_{j^\dagger}^{(s)}}{B_{j^\dagger}}\right) \cdot f_s\left(\frac{\hat{y}_{j^\dagger}^{(s+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} + \left(\frac{\hat{y}_{j^\dagger}^{(k+1)} - \hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}}\right) \cdot f_k\left(\frac{y_{j^\dagger}^{(k+1)}}{B_{j^\dagger}}\right) w_{j^\dagger} + \frac{w_{j^\dagger}}{B_{j^\dagger}} \\
\leq &\sum_{s=1}^{k-1} \left(\frac{\hat{y}_{j^\dagger}^{(s+1)} - \hat{y}_{j^\dagger}^{(s)}}{B_{j^\dagger}}\right) \cdot \left(f_s\left(\frac{\hat{y}_{j^\dagger}^{(s+1)} - 1}{B_{j^\dagger}}\right) + \frac{1}{B_{j^\dagger}}\right) w_{j^\dagger} \\
&\qquad + \left(\frac{\hat{y}_{j^\dagger}^{(k+1)} - \hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}}\right) \cdot \left(f_k\left(\frac{\hat{y}_{j^\dagger}^{(k+1)} - 1}{B_{j^\dagger}}\right) + \frac{2}{B_{j^\dagger}}\right) w_{j^\dagger} + \frac{w_{j^\dagger}}{B_{j^\dagger}} \\
= &\sum_{s=1}^{k} \Delta\beta_{j^\dagger}^{(s)} + \frac{\hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}} \frac{w_{j^\dagger}}{B_{j^\dagger}} + \frac{\hat{y}_{j^\dagger}^{(k+1)} - \hat{y}_{j^\dagger}^{(k)}}{B_{j^\dagger}} \frac{2w_{j^\dagger}}{B_{j^\dagger}} + \frac{w_{j^\dagger}}{B_{j^\dagger}} \leq \sum_{s=1}^{k} \Delta\beta_{j^\dagger}^{(s)} + \frac{3w_{j^\dagger}}{B_{j^\dagger}}
\end{aligned}$$

where the first inequality uses the fact that $f_k(a) \leq 1$ for all $a \in [0,1]$ and $\hat{y}_{j^\dagger}^{(k+1)} \geq y_{j^\dagger}^{(k+1)} - 1$ by construction; the second inequality uses the fact that $f_k'(a) \in [0,1]$ for all $a \in [0,1]$.

In $k = K$ case, consider any edge $(i^\dagger, j^\dagger) \in E_K$. Notice that $\alpha_{i^\dagger} + \Delta\beta_{j^\dagger}^{(K)} \geq w_j \left(1 - \frac{\hat{y}_{j^\dagger}^{(K)}}{B_{j^\dagger}}\right)$ by construction. If $\hat{y}_{j^\dagger}^{(K)} = 0$, inequality (2) holds immediately. Otherwise, let $k' < K$ be the largest stage such that $\hat{y}_{j^\dagger}^{(k')} < \hat{y}_{j^\dagger}^{(k'+1)} = \hat{y}_{j^\dagger}^{(K)}$. By construction, we know that $\Delta\beta_{j^\dagger}^{(s)} = 0$ for all $s \in [k'+1, K-1]$. Note that

$$\begin{aligned}
\alpha_i + \sum_{s=1}^{K} \Delta\beta_{j^\dagger}^{(s)} &\geq w_j \left(1 - \frac{\hat{y}_{j^\dagger}^{(K)}}{B_{j^\dagger}}\right) + \sum_{s=1}^{K-1} \Delta\beta_{j^\dagger}^{(s)} \\
&= w_j \left(1 - \frac{\hat{y}_{j^\dagger}^{(k'+1)}}{B_{j^\dagger}}\right) + \sum_{s=1}^{k'} \Delta\beta_{j^\dagger}^{(s)} \\
&\geq w_j \left(1 - f_{k'}\left(\frac{\hat{y}_{j^\dagger}^{(k'+1)}}{B_{j^\dagger}}\right)\right) + \sum_{s=1}^{k'} \Delta\beta_{j^\dagger}^{(s)} \\
&\geq \left(1 - (1 - 1/K)^K\right) w_{j^\dagger} - \frac{3w_j}{B_{j^\dagger}}
\end{aligned}$$

where the third inequality uses the fact that $f_s(a) \geq a$ for all $a \in [0,1]$ and $s \in [K-1]$; and the fourth inequality uses the similar argument as in $k \in [K-1]$ case. □

REMARK 6 (**Optimality of the $\Gamma(K)$ Competitive Ratio for b-matching**). Note that the multi-stage vertex weighted b-matching problem generalizes the multi-stage (fractional) vertex weighted matching problem. Therefore Theorem 3 implies that the competitive ratio in Proposition 3 has the optimal dependence on K in the multi-stage vertex weighted b-matching problem, i.e., no integral or fractional multi-stage algorithm can obtain a competitive ratio better than $\Gamma(K)$.

B.2. AdWords Problem

In this subsection, we consider the K-stage AdWords problem.

Model Description The multi-stage AdWords problem is a variant of the multi-stage weighted matching where (i) the weights (a.k.a., bids) $\{b_{ij}\}_{(i,j)\in E}$ are assigned to edges; (ii) the matching constraint of each offline vertex $j \in V$ is replaced with the budget constraint B_j; and the goal is to maximize the total bids (a.k.a., used budgets) in the allocation. Let $B_{\min} = \min_{j\in V} B_j$ be the smallest budget.

Connection to Configuration Allocation The AdWords problem is a special case of the configuration allocation. To see this, we consider a reduction in which every instance in the AdWords problem can be thought as an instance in the configuration allocation as follows: Each advertiser j has a demand $n_j \leftarrow B_j$. The feasible configuration set encodes all edges E in the AdWords problem ($C \leftarrow E$). Moreover, for every edge $(i^\dagger, j^\dagger) \in E$, the corresponding configuration c encodes the number of reserved impression $\{\xi_{i,c,j}\}_{i,j}$ as $\xi_{i,c,j} \leftarrow b_{ij} \cdot \mathbb{1}\{i = i^\dagger, j = j^\dagger\}$. The per-impression prices are identical for each advertiser j ($w_{i,c,j} \leftarrow \mathbb{1}\{(i,j) \in E\}$). Finally, note that since per-impression price are identical for each advertiser and number of reserved impression $\{\xi_{i,c,j}\}$ is encoded as the aforementioned indicator function, without loss of generality, we can restrict attention to online algorithms whose allocations never exceed demand $\{n_j\}$ of advertisers, and thus no preemption or free disposal is needed.

Fractional Multi-stage AdWords Given the connection to the configuration allocation problem, we can implement Algorithm 2 where the convex program $\mathcal{P}_k^{\text{MCA}}$ can be simplified as program $\mathcal{P}_k^{\text{AdWords}}$ defined below,

$$\mathbf{z}^{(k)} = \arg\max_{\mathbf{z}} \sum_{(i,j)\in E_k} b_{ij} z_{ij} - \sum_{j\in V} B_j F_k\left(\frac{y_j}{B_j} + \sum_{i:(i,j)\in E_k} \frac{b_{ij} z_{ij}}{B_j}\right) \text{ s.t.}$$

$$\begin{aligned}
&\sum_{j:(i,j)\in E_k} z_{ij} \le 1 && i \in U_k \\
&\sum_{i:(i,j)\in E_k} b_{ij} z_{ij} \le B_j - y_j && j \in V \\
&z_{ij} \ge 0 && (i,j) \in E_k
\end{aligned} \tag{$\mathcal{P}_k^{\text{AdWords}}$}$$

Since the feasible configuration sets encodes all edges E in the AdWords problem, variables $\mathbf{x}$ in program $\mathcal{P}_k^{\text{MCA}}$ become redundant and are removed in program $\mathcal{P}_k^{\text{VWM}}$. Moreover, since there is no need for preemption in the AdWords problem, variables $\mathbf{y}$ in program $\mathcal{P}_k^{\text{MCA}}$ become redundant and we reuse this notation to denote the allocated budgets from previous stages in program $\mathcal{P}_k^{\text{AdWords}}$.

See the *Polynomial Regularized Fractional AdWords* (`PR-F-AdWords`) algorithm for a formal description (Algorithm 4).

Rounding to an Integral Allocation To output integral solutions in the K-stage AdWords problem, we apply a standard randomized rounding technique as follows. At each stage k, find the optimal fractional solution

Algorithm 4: Polynomial Regularized Fractional AdWords (`PR-F-AdWords`)

Input: Number of batches/stages K

$\forall j \in V:\ y_j \leftarrow 0$.

for *stage* $k=1$ *to* K **do**

`/* ` k^{th} `batch of vertices in` U`, denoted by` U_k`, arrives. Let` E_k `denote the set of edges incident to vertices in` U_k`. */`

Given $\{y_j\}_{j\in V}$ and subgraph $G_k=(U_k,V,E_k)$, solve $\mathbf{z}^{(k)}$ from the convex program $\mathcal{P}_k^{\text{AdWords}}$.

$\forall j\in V: y_j \leftarrow y_j + \sum_{i\in U_k} b_{ij} z_{ij}^{(k)}$.

Return $\mathbf{z}^{(k)}$ as the fractional AdWords allocation of stage k.

$\mathbf{z}^{(k)}$ of the convex program $\mathcal{P}_k^{\text{AdWords}}$ with adjusted budgets $\{(1-\rho)B_j\}_{j\in V}$, where $\rho = O(\sqrt{\log(B_{\min})/B_{\min}})$ for a large enough constant in the big O notation. For each online vertex $i\in U_k$, sample an offline vertex $j\in V$ independently with probability $\{z_{ij}^{(k)}\}_{j:(i,j)\in E_k}$. If the offline vertex j has enough budget for bid b_{ij}, allocate the online vertex i to the offline vertex j. Otherwise, leave the online vertex i unmatched. Finally, update $\{y_j\}_{j\in V}$ with the fractional solution $\mathbf{z}^{(k)}$ as before. We refer to this algorithm as *Polynomial Regularized Integral AdWords* (`PR-I-AdWords`). See Algorithm 5 for a formal description.

We are now ready to describe the competitive ratio results of the multi-stage AdWords problem.

Proposition 4 **(Optimal Multi-stage Algorithm for Fractional AdWords)**. *In the K-stage fractional AdWords problem, `PR-F-AdWords` (Algorithm 4) is $\Gamma(K)$-competitive, where $\Gamma(K)=1-\left(1-\frac{1}{K}\right)^K$.*

Proposition 5 **(Optimal Multi-stage Algorithm for AdWords with Large Budgets)**. *In the K-stage integral AdWords problem, `PR-I-AdWords` (Algorithm 5) is $\left(\Gamma(K)-O(\sqrt{\log(B_{min})/B_{min}})\right)$-competitive, where $\Gamma(K)=1-\left(1-\frac{1}{K}\right)^K$.*

The competitive ratio of the fractional solution in Proposition 4 is directly implied by Theorem 2 since the fractional AdWords problem is a special case of the fractional configuration allocation problem. Below we show the proof of the competitive ratio of the integral solution in Proposition 5 using a simple rounding argument.

Proof of Proposition 5. Let $\{z_{ij}^{(k)}\}_{k\in[K],(i,j)\in E_k}$ be the fractional solution generated in Algorithm 5, i.e., the optimal solution of the program $\mathcal{P}_k^{\text{AdWords}}$ with adjusted budgets $\{(1-\rho)B_j\}_{j\in V}$. Based on Proposition 4 for fractional solutions, we know that fractional solution $\{z_{ij}^{(k)}\}_{k\in[K],(i,j)\in E_k}$ is a $\Gamma(K)$-approximation to the optimal offline with adjusted budgets $\{(1-\rho)B_j\}_{j\in V}$. Thus, $\{z_{ij}^{(k)}\}_{k\in[K],(i,j)\in E_k}$ is a $(1-\rho)\Gamma(K)$-approximation to the optimal offline with original budgets $\{B_j\}_{j\in V}$.

Let $\{\hat{z}_{ij}^{(k)}\}_{k\in[K],(i,j)\in E_k}$ be the integral solution generated in Algorithm 5. For any stage $k\in[K]$, any edge $(i,j)\in E_k$, note that

$$\mathbf{Pr}\left[\hat{z}_{ij}^{(k)}=1\right] = \mathbf{Pr}[\text{vertex } j \text{ is sampled for vertex } i]\cdot\mathbf{Pr}[\text{there is enough budget for bid } b_{ij}].$$

where $\mathbf{Pr}[\text{vertex } j \text{ is sampled for vertex } i] = z_{ij}^{(k)}$ by construction. Applying the multiplicative form of Chernoff bound (Lemma 6), we can lower-bound $\mathbf{Pr}[\text{there is enough budget for bid } b_{ij}]$ by $1-\exp\left(-\frac{\rho^2 B_{\min}}{2-\rho}\right)$. Thus, the

Algorithm 5: Polynomial Regularized Integral AdWords (`PR-I-AdWords`)

Input: Number of batches/stages K, parameter $\rho \in [0,1]$

$\forall j \in V:\ y_j \leftarrow 0, \hat{y}_j \leftarrow 0$.

for *stage* $k=1$ *to* K **do**

`/*` k^{th} `batch of vertices in` U`, denoted by` U_k`, arrives. Let` E_k `denote the set of edges incident to vertices in` U_k`. */`

Given $\{y_j\}_{j\in V}$ and subgraph $G_k=(U_k,V,E_k)$, solve $\mathbf{z}^{(k)}$ from the convex program $\mathcal{P}_k^{\text{AdWords}}$ with adjusted budgets $\{(1-\rho)B_j\}_{j\in V}$, and used budgets $\{y_j\}_{j\in V}$.

$\hat{\mathbf{z}}^{(k)} \leftarrow \mathbf{0}$.

for *online vertex* $i \in U_k$ **do**

Sample offline vertex $j \sim \{z_{ij}^{(k)}\}_{j:(i,j)\in E_k}$.

if $\hat{y}_j + b_{ij} \le B_j$ **then**

$\hat{z}_{ij}^{(k)} \leftarrow 1$.

$\hat{y}_j \leftarrow \hat{y}_j + b_{ij}$.

$\forall j \in V: y_j \leftarrow y_j + \sum_{i\in U_k} b_{ij} z_{ij}^{(k)}$.

Return $\hat{\mathbf{z}}^{(k)}$ as the integral AdWords allocation of stage k.

integral solution $\{\hat{z}_{ij}^{(k)}\}_{k\in[K],(i,j)\in E_k}$ is $(1-\rho)(1-\exp\left(-\frac{\rho^2 B_{\min}}{2-\rho}\right))$-approximation to the optimal offline if we consider the expected objective value. Finally, setting $\rho = O\left(\sqrt{\frac{\log(B_{\min})}{B_{\min}}}\right)$ for a large enough constant in the big O notation finishes the proof. □

LEMMA 6 **(Multiplicative Chernoff Bound, Chernoff et al., 1952)**. *Suppose $X_1, X_2, \ldots, X_t$ are independent random variables taking values in $\{0,1\}$. Let X denote their sum and let $\mu = \mathbf{E}[X]$ denote the sum's expected value. Then for any $\delta > 0$,*

$$\mathbf{Pr}[X > (1+\delta)\mu] \le \exp\left(-\frac{\delta^2\mu}{2+\delta}\right)$$

REMARK 7 (**Optimality of the $\Gamma(K)$ Competitive Ratio for AdWords**). Again, since the multi-stage AdWords problem generalizes the multi-stage (fractional) unweighted matching problem (i.e., when $b_{ij} \in \{0,1\}$ and all budgets are $B_j = 1$), Theorem 3 implies that the competitive ratios in Theorems 4 and 5 have the optimal dependence on K in the multi-stage AdWords problem, i.e., no integral or fractional multi-stage algorithm can obtain a competitive ratio better than $\Gamma(K)$.

B.3. Assortment Problem

In this subsection, we consider the K-stage assortment problem.

Model Description In the multi-stage assortment problem, each node j in V corresponds to a product with capacity B_j and reward w_j. Each node i in U corresponds a consumer with a choice model $\mathcal{C}_i : 2^V \times V \to [0,1]$.

By displaying an *assortment* $S \subseteq V$ (i.e., a subset of products) to her, consumer i selects each product $j \in S$ with probability $\mathcal{C}_i(S,j)$. If product j has available units, the platform decides whether to approval the transaction of product j to consumer i. Approving this transaction, the platform gains reward w_j and the inventory of product j is decreased by one. The goal of the platform is to maximize the expected total rewards.[19]

We introduce the *fluid relaxation* of the assortment problem as follows: for each consumer i, the platform decide a fractional assortment assignment $\{z_{iS}\}_S$ such that $\sum_S z_{iS} \le 1$ and fractional product allocation $\{x_{iSj}\}_{S,j}$ such that $x_{iSj} \le \mathcal{C}_i(S,j) \cdot z_{iS}$. Given fractional decision $\{z_{iS}, x_{iSj}\}_{S,j}$, the platform collect rewards $\sum_S \sum_j w_j x_{iSj}$ from consumer i, and the inventory of each product j is decreased by $\sum_S x_{iSj}$.

Connection to Configuration Allocation It is straightforward to verify that the fluid relaxation of the assortment problem is a special case of the configuration allocation. To see this, we consider a reduction in which every instance in the assortment problem can be thought as an instance in the configuration allocation as follows: Each advertiser j has a demand $n_j \leftarrow B_j$. The feasible configuration set encodes all assortments 2^V in the assortment problem ($C \leftarrow 2^V$). Moreover, for every assortment $S \subseteq V$, the corresponding configuration c encodes the number of reserved impression $\{\xi_{i,c,j}\}_{i,j}$ as $\xi_{i,c,j} \leftarrow \mathcal{C}_i(S,j)$. The per-impression prices are identical for each advertiser j ($w_{i,c,j} \leftarrow w_j$). Finally, note that since per-impression price are identical for each advertiser, without loss of generality, we can restrict attention to online algorithms whose allocations never exceed demand $\{n_j\}$ of advertisers, and thus no preemption or free disposal is needed.

Fluid Multi-stage Assortment Given the connection to the configuration allocation problem, we can implement Algorithm 2 where the convex program $\mathcal{P}_k^{\text{MCA}}$ can be simplified as program $\mathcal{P}_k^{\text{assortment}}$ defined below,

$$\mathbf{x}^{(k)}, \mathbf{z}^{(k)} = \arg\max_{\mathbf{x},\mathbf{z} \ge \mathbf{0}} \sum_{i \in U_k} \sum_{S \subseteq V} \sum_{j \in S} w_j x_{iSj} - \sum_{j \in V} B_j F_k \left(\frac{y_j}{B_j} + \sum_{i \in U_k} \sum_{S \subseteq V : S \ni j} \frac{x_{iSj}}{B_j} \right) \text{ s.t.}$$

$$\begin{aligned}
&\sum_{S \subseteq V} z_{iS} \le 1 && i \in U_k \\
&x_{iSj} \le \mathcal{C}_i(S,j) \cdot z_{iS} && i \in U_k, S \subseteq V, j \in S \\
&\sum_{i \in U_k} \sum_{S \subseteq V : S \ni j} x_{iSj} \le B_j - y_j && j \in V
\end{aligned} \qquad (\mathcal{P}_k^{\text{assortment}})$$

Since there is no need for preemption in the assortment problem, variables $\mathbf{y}$ in program $\mathcal{P}_k^{\text{MCA}}$ become redundant and we reuse this notation to denote the allocated inventory from previous stages in program $\mathcal{P}_k^{\text{AdWords}}$.

See the *Polynomial Regularized Fractional Assortment* (`PR-F-Assortment`) algorithm for a formal description (Algorithm 6).

Rounding to an Integral Allocation To output integral solutions in the K-stage assortment problem, we apply a standard randomized rounding technique (similar to the one for AdWords problem) as follows. At each stage k, find the optimal fractional solution $(\mathbf{x}^{(k)}, \mathbf{z}^{(k)})$ of the convex program $\mathcal{P}_k^{\text{assortment}}$ with adjusted inventory $\{(1-\rho)B_j\}_{j \in V}$, where $\rho = O(\sqrt{\log(B_{\min})/B_{\min}})$ for a large enough constant in the big O notation. For each

[19] For simplicity, we study the model where the platform can display an assortment with non-available products and has the ability to revoke consumers' selections. Our result can be extended to the alternative model where the platform are only allowed to display assortments with available products and has no revoking power, using the sub-assortment sampling technique developed in Feng et al. (2022).

Algorithm 6: Polynomial Regularized Fluid Assortment (`PR-F-Assortment`)

Input: Number of batches/stages K

$\forall j \in V:\ y_j \leftarrow 0$.

for *stage* $k=1$ *to* K **do**

  `/*` k^{th} `batch of consumers in` U`, denoted by` U_k`, arrives. Let` $\{\mathcal{C}_i\}_{i\in U_k}$ `denote the choice models of consumers in` U_k`. */`

  Given $\{y_j\}_{j\in V}$ and consumers U_k with choice models $\{\mathcal{C}_i\}_{i\in U_k}$, solve $\mathbf{x}^{(k)}, \mathbf{z}^{(k)}$ from the convex program $\mathcal{P}_k^{\text{assortment}}$.

  $\forall j \in V: y_j \leftarrow y_j + \sum_{i\in U_k} \sum_{S\subseteq V: S\ni j} x_{iSj}^{(k)}$.

  Return $(\mathbf{x}^{(k)}, \mathbf{z}^{(k)})$ as the fractional assortment allocation of stage k.

online consumer $i \in U_k$, sample an assortment $S \subseteq V$ independently with probability $\{z_{iS}^{(k)}\}_S$. Suppose consumer i selects product $j \in S$ and product j has available units. Approve the transaction of product j to consumer i with probability $\frac{x_{iSj}^{(k)}}{\mathcal{C}_i(S,j)\cdot z_{iS}^{(k)}}$. Finally, update $\{y_j\}_{j\in V}$ with the fractional solution $\mathbf{x}^{(k)}$ as before. We refer to this algorithm as *Polynomial Regularized Integral Assortment* (`PR-I-Assortment`). See Algorithm 7 for a formal description.

We are now ready to describe the competitive ratio results of the multi-stage assortment problem.

Proposition 6 **(Optimal Multi-stage Algorithm for Fluid Assortment)**. *In the K-stage fluid Assortment problem, `PR-F-Assortment` (Algorithm 6) is $\Gamma(K)$-competitive, where $\Gamma(K) = 1-\left(1-\frac{1}{K}\right)^K$.*

Proposition 7 **(Optimal Multi-stage Algorithm for Assortment with Large Inventories)**. *In the K-stage integral assortment problem, `PR-I-Assortment` (Algorithm 7) is $\left(\Gamma(K) - O(\sqrt{\log(B_{min})/B_{min}})\right)$-competitive, where $\Gamma(K) = 1-\left(1-\frac{1}{K}\right)^K$.*

The competitive ratio of the fluid solution in Proposition 6 is directly implied by Theorem 2 since the fluid assortment problem is a special case of the fractional configuration allocation problem. The competitive ratio of the integral solution in Proposition 7 follows the same argument as the one in Proposition 5 and thus is omitted here.

Remark 8 (**Optimality of the $\Gamma(K)$ Competitive Ratio for Assortment**). Again, since the multi-stage assortment problem generalizes the multi-stage (fractional) unweighted matching problem (i.e., when $\mathcal{C}_i(S,i) = \mathbb{1}\{S = |i|\}$ and all inventories are $B_j = 1$), Theorem 3 implies that the competitive ratios in Theorems 6 and 7 have the optimal dependence on K in the multi-stage assortment problem, i.e., no integral or fractional multi-stage algorithm can obtain a competitive ratio better than $\Gamma(K)$.

C. Missing Proofs

C.1. Proof of Proposition 1

We first prove (i) by backward induction on k:

Algorithm 7: Polynomial Regularized Integral Assortment (`PR-I-Assortment`)

Input: Number of batches/stages K, parameter $\rho \in [0,1]$

$\forall j \in V:\ y_j \leftarrow 0, \hat{y}_j \leftarrow 0$.

for *stage* $k=1$ *to* K **do**

`/* ` k^{th} `batch of consumers in` U`, denoted by` U_k`, arrives. Let` $\{\mathcal{C}_i\}_{i\in U_k}$ `denote the choice models of consumers in` U_k`. */`

Given $\{y_j\}_{j\in V}$ and consumers U_k with choice models $\{\mathcal{C}_i\}_{i\in U_k}$, solve $\mathbf{x}^{(k)}, \mathbf{z}^{(k)}$ from the convex program $\mathcal{P}_k^{\text{assortment}}$ with adjusted inventories $\{(1-\rho)B_j\}_{j\in V}$, and used inventories $\{y_j\}_{j\in V}$.

for *online consumer* $i \in U_k$ **do**

Sample assortment $S \sim \{z_{iS}^{(k)}\}_{S\subseteq V}$.

Display assortment S to consumer i.

if *Consumer* i *selects product* j *and* $\hat{y}_j + 1 \le B_j$ **then**

Approve transaction of product j to consumer i with probability $\frac{x_{iSj}^{(k)}}{\mathcal{C}_i(S,j)\cdot z_{iS}^{(k)}}$.

$\forall j \in V: y_j \leftarrow y_j + \sum_{i\in U_k}\sum_{S\subseteq V: S\ni j} x_{iSj}^{(k)}$.

- *Base case* ($k=K-1$)*:* The maximum of function $(1-y)f_K(x+y) = (1-y)\mathbb{I}\{x+y=1\}$ over $y\in[0,1-x]$ happens at $y^*=1-x$, and therefore $f_{K-1}(x)=x$.
- *Inductive step* ($k\in[K-2]$). Suppose $f_{k+1}(x) = (1-\frac{1-x}{K-k-1})^{K-k-1}$. Fix $x\in[0,1]$ and let $y^* = \arg\max_{y\in[0,1-x]} (1-y)f_{k+1}(x+y)$. By writing the first-order condition due to optimality of y^*, we have:
$$\left(1-\frac{1-x-y^*}{K-k-1}\right)^{K-k-2} - y^*\left(1-\frac{1-x-y^*}{K-k-1}\right)^{K-k-2} - \left(1-\frac{1-x-y^*}{K-k-1}\right)^{K-k-1} = 0\ , \tag{6}$$
which implies that $y^* = \frac{1-x}{K-k}$. Note that $y^*\in[0,1-x]$. To check whether y^* is the maximum point in $[0,1-x]$, we consider the second-order condition with respect to y^*. After algebraic simplifications, we have (details are omitted for brevity):
$$\frac{\partial^2}{\partial^2 y}\left[(1-y)f_{k+1}(x+y)\right]_{y=y^*} = (x-2)(1-\frac{1-x-y^*}{K-k-1})^{K-k-3} \le 0$$
Plugging back $y^* = \frac{1-x}{K-k}$ into the objective, we have:
$$f_k(x) = (1-\frac{1-x}{K-k})f_{k+1}(x+\frac{1-x}{K-k}) = (1-\frac{1-x}{K-k})(1-\frac{1-x}{K-k})^{K-k-1} = (1-\frac{1-x}{K-k})^{K-k}\ ,$$
as desired.

Given (i), proof of (ii) is immediate as the polynomial $f_k(x) = (1-\frac{1-x}{K-k})^{K-k}$ is monotone increasing for $k\in[K-1]$. Also, (iii) simply holds as $(1-x)\le e^{-x}$ when $x\in[0,1]$, and that $\lim_{N\to+\infty}(1-\frac{x}{N})^N = e^{-x}$ for $x\in[0,1]$. Finally, to prove (iv), consider defining $f_0(x) = \max_{y\in[0,1-x]}(1-y)f_1(x+y)$. Following the backward induction in the proof of (i), $f_0(x) \triangleq (1-(1-x)/K)^K$. Therefore,
$$\min_{y\in[0,1]} 1-(1-y)f_1(y) = 1-\max_{y\in[0,1]}(1-y)f_1(y) = 1-f_0(0) = \Gamma(K).$$

C.2. Proof of Proposition 2

We first define the Lagrangian dual function $\mathcal{L}_k(\cdot)$ and then re-state the KKT optimality conditions of $\mathbf{x}_k^*$ in our specific convex program.

DEFINITION 2. The *Lagrangian dual* $\mathcal{L}_k(.)$ for the concave program $\mathcal{P}_k^{\text{VWM}}$ is defined as:

$$\begin{aligned}\mathcal{L}_k(\mathbf{x},\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) \triangleq & \sum_{j\in V} w_j \left(\sum_{i:(i,j)\in E_k} x_{ij} - F_k\left(y_j + \sum_{i:(i,j)\in E_k} x_{ij}\right)\right) \\ & - \sum_{i\in U_k} \lambda_i \left(\sum_{j:(i,j)\in E_k} x_{ij} - 1\right) - \sum_{j\in V} \theta_j \left(\sum_{i:(i,j)\in E_k} x_{ij} + y_j - 1\right) \\ & + \sum_{(i,j)\in E_k} \gamma_{ij} x_{ij} \quad ,\end{aligned}$$

where $\{\lambda_i\}$, $\{\theta_j\}$, and $\{\gamma_{ij}\}$ are the *Lagrange multipliers*, also known as the *dual variables*, corresponding to different constraints in program $\mathcal{P}_k^{\text{VWM}}$.

PROPOSITION 8 **(KKT Conditions for Multi-stage Polynomial Regularized Matching)**. *Suppose* $\mathbf{x}^*$ *is the optimal solution of the concave program* $\mathcal{P}_k^{VWM}$*. Consider the Lagrangian dual* $\mathcal{L}_k$*, as in Definition 2. Then:*

$$\max_{\mathbf{x}} \min_{\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}} \mathcal{L}_k(\mathbf{x},\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) = \min_{\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}} \max_{\mathbf{x}} \mathcal{L}_k(\mathbf{x},\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) = \min_{\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}} \mathcal{L}_k(\mathbf{x}^{(k)},\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) \quad .$$

Moreover, if $\boldsymbol{\lambda}^*$*,* $\boldsymbol{\theta}^*$*, and* $\boldsymbol{\gamma}^*$ *(together with* $\mathbf{x}^*$*) are the solutions of the above minmax/maxmin, then:*

- *(Stationarity)* $\forall (i,j)\in E_k:\ \frac{\partial \mathcal{L}_k}{\partial x_{ij}}(\mathbf{x}^*,\boldsymbol{\lambda}^*,\boldsymbol{\theta}^*,\boldsymbol{\gamma}^*) = 0,$
- *(Dual feasibility)* $\forall i\in U_k, j\in V, (i,j)\in E_k:\ \lambda_i^*\geq 0,\ \theta_j^*\geq 0,$ *and* $\gamma_{ij}^*\geq 0$,
- *(Complementary slackness)* $\forall i\in U_k, j\in V, (i,j)\in E_k:$
  - $\gamma_{ij}^* = 0$ *or* $\left(\gamma_{ij}^* > 0 \text{ and } x_{ij}^* = 0\right)$,
  - $\lambda_i^* = 0$ *or* $\left(\lambda_i^* > 0 \text{ and } \sum_{j:(i,j)\in E_k} x_{ij}^* = 1\right)$,
  - $\theta_j^* = 0$ *or* $\left(\theta_j^* > 0 \text{ and } y_j + \sum_{i:(i,j)\in E_k} x_{ij}^* = 1\right)$.

With the above ingredients, we prove the structural lemma.

Proof of Proposition 2. We first show the uniformity property. For $j\in V_{k,0}$, note that

$$w_j\left(1 - f_k\left(y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)}\right)\right) = w_j(1-f_k(1)) = 0 = c_0^{(k)}.$$

For vertices in $V_{k,\ell}$ with $\ell\in[L_k]$, since they are in the same connected component of $G_k'[U_k\setminus U_{k,0}, V\setminus V_{k,0}]$, it is sufficient to restrict attention to $j,j'\in V_{k,\ell}$ such that there exists a vertex $i\in U_{k,\ell}$ such that $x_{ij}^{(k)}>0$ and $x_{ij'}^{(k)}>0$ (and therefore, $\gamma_{ij}^* = \gamma_{ij'}^* = 0$, because of the complementary slackness in Proposition 8). Also, $\theta_j^* = \theta_{j'}^* = 0$, again because of complementary slackness and the fact that $y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)} < 1$ and $y_{j'} + \sum_{i:(i,j')\in E_k} x_{ij'}^{(k)} < 1$. Now, applying the stationarity condition in Proposition 8, we have:

$$w_j\left(1 - f_k\left(y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)}\right)\right) - \lambda_i^* - \theta_j^* + \gamma_{ij}^* = 0 = w_{j'}\left(1 - f_k\left(y_{j'} + \sum_{i:(i,j')\in E_k} x_{ij'}^{(k)}\right)\right) - \lambda_i^* - \theta_{j'}^* + \gamma_{ij'}^* \ ,$$

and therefore $w_j\left(1 - f_k\left(y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)}\right)\right) = w_{j'}\left(1 - f_k\left(y_{j'} + \sum_{i:(i,j')\in E_k} x_{ij'}^{(k)}\right)\right)$, finishing the proof of the uniformity property.

To show saturation property, note that $\lambda_i^* = w_j \left(1 - f_k\left(y_j + \sum_{i:(i,j)\in E_k} x_{ij}^{(k)}\right)\right) > 0$ for any $i \in U_{k,\ell}, j \in V_{k,\ell}$ with $\ell \in [L_k]$ such that $x_{ij}^{(k)} > 0$, simply because $f_k(1) = 1$ and $f_k(x)$ is (strictly) monotone increasing. Again, by using complementary slackness, we have $\sum_{j:(i,j)\in E_k} x_{ij}^{(k)} = 1$, which proves the saturation property.

Now suppose there exists an edge $(i, j') \in E_k$, where $i \in U_{k,\ell}$ and $j' \in V_{k,\ell'}$, for $\ell, \ell' \in [0 : L_k], \ell \neq \ell'$. First note that there should exist a vertex $j \in V_{k,\ell}$ so that $x_{ij}^{(k)} > 0$; this holds because i is either fully matched by $\mathbf{x}^{(k)}$ if $\ell \in [L_k]$ – thanks to the saturation property – or $l = 0$ and i is connected to a vertex $j \in V_{k,0}$ with $x_{ij'}^{(k)} > 0$ by definition. Because of complementary slackness $\gamma_{ij}^* = 0$. Now, by writing the stationarity condition of Proposition 8 for the edge (i, j'), and noting that $w_{j'} \left(1 - f_k\left(y_{j'} + \sum_{i:(i,j')\in E_k} x_{ij'}^{(k)}\right)\right) = c_{\ell'}^{(k)}$ because of the uniformity property, we have:

$$c_{\ell'}^{(k)} - \lambda_i^* - \theta_{j'}^* + \gamma_{ij'}^* = 0 \ ,$$

By writing the same condition for the edge (i, j) we have:

$$c_{\ell}^{(k)} - \lambda_i^* - \theta_j^* = 0 \ ,$$

and therefore

$$c_{\ell}^{(k)} \geq c_{\ell}^{(k)} - \theta_j^* = c_{\ell'}^{(k)} - \theta_{j'}^* + \gamma_{ij'}^* \geq c_{\ell'}^{(k)} - \theta_{j'}^*$$

If $\ell' = 0$, then clearly $c_{\ell}^{(k)} \geq 0 = c_{\ell'}^{(k)}$. If $\ell' \in [L_k]$, then $\sum_{i':(i',j')\in E_k} x_{i'j'}^{(k)} + y_{j'} < 1$ and $\theta_{j'}^* = 0$ because of complementary slackness. Therefore, $c_{\ell}^{(k)} \geq c_{\ell'}^{(k)}$, which finishes the proof of the monotonicity property. □

C.3. Proof of Lemma 2

Let $(\boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\mu}^*, \boldsymbol{\pi}^*, \boldsymbol{\psi}^*, \boldsymbol{\phi}^*, \boldsymbol{\iota}^*)$ be the optimal solution in the Lagrangian dual program $\mathcal{D}_k^{\text{MCA}}$. The property complementary slackness is satisfied by considering the complementary slackness KKT conditions. stationarity-1 is due to the stationarity/first-order KKT condition with respect to $z_{i,c}$. To show stationarity-2, note that the stationarity/first-order KKT condition with respect to $x_{i,c,j}$ implies that

$$w_{i,c,j} - \sum_{\tau\in[T]:\hat{w}(\tau)\leq w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) - \theta_j^* - \mu_{i,c,j}^* + \psi_{i,c,j}^* = 0$$

Thus, it is sufficient to show that $\theta_j^* = 0$ for every advertiser j. By complementary slackness in KKT conditions, we know that $\theta_j^* = 0$ if $H_{j,0}(\mathbf{x}^*, \mathbf{y}^*)$ $(\equiv \sum_{i\in U_k} \sum_{c\in C} x_{i,c,j}^* + \sum_{\tau\in[T]} y_{j,\tau}^*)$ < 1.

Now we consider any advertiser j where $H_{j,0}(\mathbf{x}^*, \mathbf{y}^*) = 1$. Let $\tau^\dagger$ be the smallest index τ such that (i) $y_{j,\tau^\dagger}^* > 0$ or (ii) there exists $x_{i,c,j}^* > 0$ with $w_{i,c,j} = \hat{w}_{\tau^\dagger}$. Note that by definition, for any $\tau \leq \tau^\dagger$,

$$f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) = f_k(1) = 1 \tag{7}$$

We use similar arguments for both case (i) and case (ii).

- Suppose $y_{j,\tau^\dagger}^* > 0$. We have that

$$\begin{aligned} 0 &\overset{(a)}{=} \hat{w}_{\tau^\dagger} - \sum_{\tau\leq\tau^\dagger} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) - \theta_j^* - \pi_{j,\tau^\dagger}^* + \iota_{j,\tau^\dagger}^* \\ &\overset{(b)}{=} \hat{w}_{\tau^\dagger} - \sum_{\tau\leq\tau^\dagger} (\hat{w}_\tau - \hat{w}_{\tau-1}) - \theta_j^* - \pi_{j,\tau^\dagger}^* \overset{(c)}{\leq} -\theta_j^* \overset{(d)}{\leq} 0 \end{aligned}$$

  where equation (a) is due to the stationarity/first-order KKT condition with respect to $y_{j,\tau^\dagger}$; equation (b) is due to the complementary slackness KKT condition (i.e., $\iota_{j,\tau^\dagger}^* \cdot y_{j,\tau^\dagger}^* = 0$) and equation (7); and inequality (c) and (d) is due to the non-negativity of θ_j^* and $\pi_{j,\tau^\dagger}^*$.

- Similarly, suppose there exists $x^*_{i,c,j} > 0$ with $w_{i,c,j} = \hat{w}_{\tau^\dagger}$. Applying the stationarity/first-order KKT condition with respect to $x^*_{i,c,j}$ and invoking equation (7), we have

$$\begin{aligned} 0 &= w_{i,c,j} - \sum_{\tau \in [T]: \hat{w}_\tau \le w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) - \theta^*_j - \mu^*_{i,c,j} + \psi^*_{i,c,j} \\ &= \hat{w}_{\tau^\dagger} - \sum_{\tau \le \tau^\dagger} (\hat{w}_\tau - \hat{w}_{\tau-1}) - \theta^*_j - \mu^*_{i,c,j} \le -\theta^*_j \le 0 \end{aligned}$$

  Thus, we conclude that $\theta^*_j = 0$ for all advertiser j, and finish the proof of property stationarity-2. □

C.4. Proof of Lemma 3

Let $(\boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\mu}^*, \boldsymbol{\pi}^*, \boldsymbol{\psi}^*, \boldsymbol{\phi}^*, \boldsymbol{\iota}^*)$ be the optimal solution in the Lagrangian dual program $\mathcal{D}^{\text{MCA}}_k$. Suppose $\eta^{(k-1)}_{j,\tau} - y^*_{j,\tau} > 0$ for $j \in V$ and $\tau \in [T]$. The stationarity/first-order KKT condition with respect to $y_{j,\tau}$ implies that

$$\hat{w}_\tau - \sum_{\tau' \le \tau} (\hat{w}_{\tau'} - \hat{w}_{\tau'-1}) \cdot f_k(H_{j,\tau'}(\mathbf{x}^*, \mathbf{y}^*)) - \theta^*_j - \pi^*_{j,\tau} + \iota^*_{j,\tau} = 0$$

Note that $\iota_{j,\tau} \ge 0$. The complementary slackness in KKT condition implies that $\pi^*_{j,\tau} = 0$ since $\eta^{(k-1)}_{j,\tau} - y^*_{j,\tau} > 0$. Additionally, as we already shown in Lemma 2, $\theta^*_j = 0$. Hence, we have

$$\hat{w}_\tau - \sum_{\tau' \le \tau} (\hat{w}_{\tau'} - \hat{w}_{\tau'-1}) \cdot f_k(H_{j,\tau'}(\mathbf{x}^*, \mathbf{y}^*)) \le 0$$

Since $f_k(a) \le 1$ for all $a \in [0,1]$ with equality hold when $a = 1$, we conclude that $f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) = 1$ and $H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*) = 1$. □

C.5. Proof of Corollary 1

To see the proof of the first item (**Distribution Monotonicity**), by definition, $H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*) \ge \sum_{\tau' \ge \tau} y^*_{j,\tau'}$. Thus it is sufficient to show the statement when $\sum_{\tau' \ge \tau} \left(\eta^{(k-1)}_{j,\tau'} - y^*_{j,\tau'} \right) > 0$. In this case, let index $\tau^\dagger$ be the index such that $\tau^\dagger \ge \tau$ and $\eta^{(k-1)}_{j,\tau^\dagger} - y^*_{j,\tau^\dagger} > 0$. By Lemma 3, we have $H_{j,\tau^\dagger}(\mathbf{x}^*, \mathbf{y}^*) = 1$. Combining with the fact that $H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*) \ge H_{j,\tau^\dagger}(\mathbf{x}^*, \mathbf{y}^*)$ and $\sum_{\tau' \ge \tau} \eta^{(k-1)}_{j,\tau'} \le 1$ finishes the proof of the first item.

For the proof of the second item (**Only Disposing Smallest**), the argument is similar to Corollary 1. Note that it is sufficient to show the statement when $\sum_{\tau' \ge \tau} \left(\eta^{(k-1)}_{j,\tau'} - y^*_{j,\tau'} \right) > 0$. In this case, let index $\tau^\dagger$ be the index such that $\tau^\dagger \ge \tau$ and $\eta^{(k-1)}_{j,\tau^\dagger} - y^*_{j,\tau^\dagger} > 0$. By Lemma 3, we have $f_k(H_{j,\tau^\dagger}(\mathbf{x}^*, \mathbf{y}^*)) = 1$. Combining with the fact that $f_k(H_{j,\tau^\dagger}(\mathbf{x}^*, \mathbf{y}^*)) \le f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) \le 1$, we conclude that $f_k(H_{j,\tau}(\mathbf{x}^*, \mathbf{y}^*)) = 1$ which finishes the proof.

□

C.6. Proof of Lemma 4

Let $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ be the optimal solution of the linear program $\mathcal{P}^{\text{MCA}}_k$ in the last stage.

Suppose $\hat{w}_\tau < \theta^*_j$. The complementary slackness (i.e., $y_{j,\tau} \cdot (\theta^*_j + \pi_{j,\tau^*} - \hat{w}_\tau) = 0$) implies that $y_{j,\tau} = 0$. Applying the complementary slackness (i.e., $\pi^*_{j,\tau} \cdot (\eta^{(K-1)}_{j,\tau} - y^*_{j,\tau}) = 0$) again, we conclude that $\eta^{(K-1)}_{j,\tau} \cdot \pi^*_{j,\tau} = 0$.

Suppose $\hat{w}_\tau > \theta^*_j$. We know that $\pi^*_{j,\tau} \ge \hat{w}_\tau - \theta^*_j > 0$. Then the complementary slackness (i.e., $\pi^*_{j,\tau} \cdot (\eta^{(K-1)}_{j,\tau} - y^*_{j,\tau}) = 0$) suggests that $\eta^{(K-1)}_{j,\tau} = y^*_{j,\tau}$. Applying the complementary slackness (i.e., $y_{j,\tau} \cdot (\theta^*_j + \pi_{j,\tau^*} - \hat{w}_\tau) = 0$), we conclude that $\eta^{(K-1)}_{j,\tau} \cdot (\hat{w}_\tau - \pi^*_{j,\tau}) = \eta^{(K-1)}_{j,\tau} \cdot \theta^*_j$.

Finally, suppose $\hat{w}_\tau = \theta^*_j$. As a sanity check, note that equality $\eta^{(K-1)}_{j,\tau} \cdot \pi^*_{j,\tau} = 0$ is equivalent to equality $\eta^{(K-1)}_{j,\tau} \cdot (\hat{w}_\tau - \pi^*_{j,\tau}) = \eta^{(K-1)}_{j,\tau} \cdot \theta^*_j$. If $y^*_{j,\tau} = 0$, following the same argument as the case $\hat{w}_\tau < \theta^*_j$ finishes the proof. Otherwise (i.e., $y^*_{j,\tau} > 0$), following the same argument as the case $\hat{w}_\tau > \theta^*_j$ finishes the proof. □

C.7. Proof of Theorem 2

Consider the suggested dual assignment in Section 4.3. We finish the proof in two steps.

[*Step i*] *Comparing objective values in primal and dual.* we first show that the objective value of the above dual assignment is equal to the objective value of the primal assignment of Algorithm 2. To show this, we consider each stage k separately. For the last stage K, this holds by construction. For each stage $k \in [K-1]$, note that by Lemma 3 the primal objective value is

$$\Delta(\text{Primal})^{(k)} = \sum_{i \in U_k} \sum_{c \in C} \sum_{j \in V} w_{i,c,j} \cdot x^{(k)}_{i,c,j} - \sum_{j \in V} \sum_{\tau \in [T]} \hat{w}_\tau \cdot \left(\eta^{(k-1)}_{j,\tau} - y^{(k)}_{j,\tau} \right)$$

To consider the dual objective value, we start with the observation that

$$\begin{aligned}
\alpha_i &= \lambda^{(k)}_i \\
&\overset{(a)}{=} \sum_{c \in C} z^{(k)}_{i,c} \cdot \lambda^{(k)}_i \\
&\overset{(b)}{=} \sum_{c \in C} z^{(k)}_{i,c} \cdot \left(\sum_{j \in V} \xi_{i,c,j} \cdot \mu^{(k)}_{i,c,j} + \phi^{(k)}_{i,c} \right) \\
&= \sum_{c \in C} \sum_{j \in V} \xi_{i,c,j} \cdot z^{(k)}_{i,c} \cdot \mu^{(k)}_{i,c,j} + \sum_{c \in C} z^{(k)}_{i,c} \cdot \phi^{(k)}_{i,c} \\
&\overset{(c)}{=} \sum_{c \in C} \sum_{j \in V} x^{(k)}_{i,c,j} \cdot \mu^{(k)}_{i,c,j} \\
&\overset{(d)}{=} \sum_{c \in C} \sum_{j \in V} x^{(k)}_{i,c,j} \cdot \left(w_{i,c,j} - \sum_{\tau \in [T]: \hat{w}(\tau) \le w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k\left(H^{(k)}_{j,\tau} \right) + \psi^{(k)}_{i,c,j} \right) \\
&\overset{(e)}{=} \sum_{c \in C} \sum_{j \in V} x^{(k)}_{i,c,j} \cdot \left(w_{i,c,j} - \sum_{\tau \in [T]: \hat{w}(\tau) \le w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k\left(H^{(k)}_{j,\tau} \right) \right)
\end{aligned}$$

Here we heavily use Lemma 2: complementary slackness for equality (a) (c) (e), stationarity-1 for equality (b), and stationarity-2 for equality (d). Thus, we can compute the dual objective as follows,

$$\Delta(\text{Dual})^{(k)} = \sum_{i \in U_k} \alpha_i + \sum_{j \in V} \Delta\beta^{(k)}_j = \sum_{i \in U_k} \sum_{c \in C} \sum_{j \in V} w_{i,c,j} \cdot x^{(k)}_{i,c,j} - \sum_{j \in V} \sum_{\tau \in [T]} \hat{w}_\tau \cdot \left(\eta^{(k-1)}_{j,\tau} - y^{(k)}_{j,\tau} \right) \equiv \Delta(\text{Primal})^{(k)}$$

[*Step ii*] *Checking approximate feasibility of dual.* First, we show that all dual assignment are non-negative. Note that the non-negativity of $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$ is guaranteed by construction. For $\boldsymbol{\beta}$, we show that $\Delta\beta^{(k)}_j \ge 0$ for all $j \in V$ and $k \in [K]$. To see this, note that for any stage $k \in [K-1]$

$$\begin{aligned}
\Delta\beta^{(k)}_j &= \sum_{i \in U_k} \sum_{c \in C} x^{(k)}_{i,c,j} \left(\sum_{\tau \in [T]: \hat{w}(\tau) \le w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k\left(H^{(k)}_{j,\tau} \right) \right) - \sum_{\tau \in [T]} \hat{w}_\tau \cdot \left(\eta^{(k-1)}_{j,\tau} - y^{(k)}_{j,\tau} \right) \\
&\overset{(a)}{=} \sum_{\tau \in [T]} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \left(\sum_{\substack{i \in U_k, c \in C: \\ w_{i,c,j} \ge \hat{w}_\tau}} x^{(k)}_{i,c,j} \right) \cdot f_k\left(H^{(k)}_{j,\tau} \right) - \sum_{\tau \in [T]} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \sum_{\tau' \ge \tau} \left(\eta^{(k-1)}_{j,\tau'} - y^{(k)}_{j,\tau'} \right) \\
&\overset{(b)}{=} \sum_{\tau \in [T]} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \left(\sum_{\substack{i \in U_k, c \in C: \\ w_{i,c,j} \ge \hat{w}_\tau}} x^{(k)}_{i,c,j} \right) \cdot f_k\left(H^{(k)}_{j,\tau} \right) - \sum_{\tau \in [T]} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \sum_{\tau' \ge \tau} \left(\eta^{(k-1)}_{j,\tau'} - y^{(k)}_{j,\tau'} \right) \cdot f_k\left(H^{(k)}_{j,\tau} \right) \\
&= \sum_{\tau \in [T]} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \left(H^{(k)}_{j,\tau} - H^{(k-1)}_{j,\tau} \right) \cdot f_k\left(H^{(k)}_{j,\tau} \right) \overset{(c)}{\ge} 0
\end{aligned} \tag{8}$$

where (a) is by changing the summation order, (b) and (c) are due to Corollary 1. For the last stage K,

$$\Delta\beta_j^{(K)} = \theta_j^{(K)} - \sum_{j\in V,\tau\in[T]} \eta_{j,\tau}^{(K-1)} \cdot (\hat{w}_\tau - \pi_{j,\tau}^{(K)}) \overset{(a)}{\geq} \theta_j^{(K)} - \sum_{j\in V,\tau\in[T]} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} \geq 0$$

where (a) uses the fact that $\theta_j^{(K)} + \pi_{j,\tau}^{(K)} \geq \hat{w}_\tau$ by definition.

Second, we show that $\alpha_i \geq \sum_{j\in V} \xi_{i,c,j} \cdot \gamma_{i,c,j}$ for all $i \in U$. This inequality is satisfied by the construction of $\boldsymbol{\alpha}$, $\boldsymbol{\gamma}$ and stationarity-1 in Lemma 2.

Finally, to finish the proof, it is sufficient for us to show that for any stage $k \in [K]$ and $(i,c,j) \in E_k$,

$$\gamma_{i,c,j} + \sum_{s=1}^{k} \Delta\beta_j^{(s)} \geq \Gamma(K) \cdot w_{i,c,j} \tag{9}$$

We show inequality (9) holds for any stage $k \in [K-1]$ and the last stage K separately.

Fix an arbitrary stage $k \in [K-1]$ and $(i,c,j) \in E_k$. Let $\tau^\dagger \in [T]$ be the index that $\hat{w}_{\tau^\dagger} = w_{i,c,j}$. Note that

$$\begin{aligned}
\gamma_{i,c,j} &= \mu_{i,c,j}^{(k)} \\
&= w_{i,c,j} - \sum_{\tau\in[T]:\hat{w}(\tau)\leq w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k\left(H_{j,\tau}^{(k)}\right) + \psi_{i,c,j}^{(k)} \\
&\geq w_{i,c,j} - \sum_{\tau\in[T]:\hat{w}(\tau)\leq w_{i,c,j}} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot f_k\left(H_{j,\tau}^{(k)}\right) \\
&= \sum_{\tau\leq\tau^\dagger} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \left(1 - f_k\left(H_{j,\tau}^{(k)}\right)\right)
\end{aligned}$$

Combining with equality (8), we have

$$\gamma_{i,c,j} + \sum_{s=1}^{k} \Delta\beta_j^{(s)} \geq \sum_{\tau\leq\tau^\dagger} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \left(1 - f_k\left(H_{j,\tau}^{(k)}\right) + \sum_{s=1}^{k} \left(H_{j,\tau}^{(k)} - H_{j,\tau}^{(k-1)}\right) \cdot f_k\left(H_{j,\tau}^{(k)}\right)\right)$$

Thus, to show the approximated dual feasibility in (9), it is sufficient to show

$$1 - f_k\left(H_{j,\tau}^{(k)}\right) + \sum_{s=1}^{k} \left(H_{j,\tau}^{(s)} - H_{j,\tau}^{(s-1)}\right) \cdot f_s\left(H_{j,\tau}^{(s)}\right) \geq \Gamma(K) \qquad \forall \tau \leq \tau^\dagger$$

which is satisfied as the property of $\{f_k\}_{k\in[K]}$ by Lemma 1 as follows.

For the last stage K, fix any $(i,c,j) \in E_K$. Let $\tau^\dagger \in [T]$ be the index such that $\hat{w}_{\tau^\dagger} = \theta_j^{(K)}$.[20] Applying Lemma 4, we have

$$\begin{aligned}
\Delta\beta_j^{(K)} &= \theta_j^{(K)} - \sum_{j\in V,\tau\in[T]} \eta_{j,\tau}^{(K-1)} \cdot (\hat{w}_\tau - \pi_{j,\tau}^{(K)}) \\
&= \theta_j^{(K)} - \sum_{\tau\geq\tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau<\tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau
\end{aligned} \tag{10}$$

We consider two subcases: $w_{i,c,j} < \theta_j^{(K)}$ and $w_{i,c,j} \geq \theta_j^{(K)}$.

[20] Note that the optimal solution of the convex program $\mathcal{P}_k^{\mathrm{MCA}}$ as well as the outcome of Algorithm 2 does not change if we enlarge the revenue level set $\{\hat{w}_\tau\}$. Thus, it is without loss of generality to assume there exists index $\tau^\dagger$ such that $\hat{w}_{\tau^\dagger} = \theta_j^{(K)}$.

Suppose $w_{i,c,j} < \theta_j^{(K)}$. Let $\tau^\ddagger$ be the index such that $\hat{w}_{\tau^\ddagger} = w_{i,c,j}$. By definition, $\tau^\ddagger < \tau^\dagger$.

$$\begin{aligned}
&\gamma_{i,c,j} + \sum_{s=1}^{K} \Delta\beta_j^{(s)} \\
\overset{(a)}{=}\ & \mu_{i,c,j}^{(K)} + \left(\theta_j^{(K)} - \sum_{\tau \geq \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau < \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
=\ & \mu_{i,c,j}^{(K)} + \left(\theta_j^{(K)} - \sum_{\tau \geq \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau=\tau^\ddagger}^{\tau^\dagger-1} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau - \sum_{\tau < \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
\overset{(b)}{\geq}\ & \mu_{i,c,j}^{(K)} + \left(\theta_j^{(K)} - \sum_{\tau \geq \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau=\tau^\ddagger}^{\tau^\dagger-1} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau < \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
\geq\ & \left(1 - \sum_{\tau \geq \tau^\ddagger} \eta_{j,\tau}^{(K-1)}\right) \cdot \left(\mu_{i,c,j}^{(K)} + \theta_j^{(K)}\right) - \sum_{\tau < \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
\overset{(c)}{\geq}\ & \left(1 - \sum_{\tau \geq \tau^\ddagger} \eta_{j,\tau}^{(K-1)}\right) \cdot w_{i,c,j} - \sum_{\tau < \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
=\ & \left(1 - H_{j,\tau^\ddagger}^{(K-1)}\right) \cdot \hat{w}_{\tau^\ddagger} - \sum_{\tau < \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
=\ & \sum_{\tau \leq \tau^\ddagger} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \left(1 - H_{j,\tau}^{(K-1)}\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
\overset{(d)}{=}\ & \sum_{\tau \leq \tau^\ddagger} (\hat{w}_\tau - \hat{w}_{\tau-1}) \cdot \left(1 - f_{K-1}\left(H_{j,\tau}^{(K-1)}\right)\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
\overset{(e)}{\geq}\ & \Gamma(K) \cdot \hat{w}_{\tau^\ddagger} = \Gamma(K) \cdot w_{i,c,j}
\end{aligned}$$

where (a) uses the construction of $\gamma_{i,c,j}$ and equality (10); (b) uses the fact that $\hat{w}_\tau \leq \theta_j^{(K)}$ for every $\tau \in [\tau^\ddagger : \tau^\dagger - 1]$; (c) uses the fact that $\mu_{i,c,j}^{(K)} + \theta_j^{(K)} \geq w_{i,c,j}$; (d) uses the fact that $f_{K-1}(a) = a$ for all $a \in [0,1]$; and (e) uses the similar argument as the one for stage $k \in [K-1]$.

Suppose $w_{i,c,j} \geq \theta_j^{(K)}$. Let $\tau^\ddagger$ be the index such that $\hat{w}_{\tau^\ddagger} = w_{i,c,j}$. By definition, $\tau^\ddagger \geq \tau^\dagger$.

$$\begin{aligned}
&\gamma_{i,c,j} + \sum_{s=1}^{K} \Delta\beta_j^{(s)} \\
=\ & \mu_{i,c,j}^{(K)} + \left(\theta_j^{(K)} - \sum_{\tau \geq \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau < \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
=\ & \mu_{i,c,j}^{(K)} + \left(\theta_j^{(K)} - \sum_{\tau \geq \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau=\tau^\dagger}^{\tau^\ddagger-1} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau < \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
\overset{(a)}{\geq}\ & \mu_{i,c,j}^{(K)} + \left(\theta_j^{(K)} - \sum_{\tau \geq \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \theta_j^{(K)} - \sum_{\tau=\tau^\dagger}^{\tau^\ddagger-1} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau - \sum_{\tau < \tau^\dagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau\right) + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)} \\
\geq\ & \left(1 - \sum_{\tau \geq \tau^\ddagger} \eta_{j,\tau}^{(K-1)}\right) \cdot \left(\mu_{i,c,j}^{(K)} + \theta_j^{(K)}\right) - \sum_{\tau < \tau^\ddagger} \eta_{j,\tau}^{(K-1)} \cdot \hat{w}_\tau + \sum_{s=1}^{K-1} \Delta\beta_j^{(s)}
\end{aligned}$$

where (a) uses the fact that $\hat{w}_\tau \geq \theta_j^{(K)}$ for every $\tau \in [\tau^\dagger : \tau^\ddagger - 1]$, and the remaining argument is identical to the subcase for $w_{i,c,j} < \theta_j^{(K)}$ which we omit here. □

D. Numerical Experiments

In this section we provide numerical justifications for the performance of our multi-stage configuration allocation algorithm on synthetic instances. Our instances are motivated by the display-ad application. We compare the performance of our algorithm with both the batched-greedy algorithm (that at each stage greedily picks the best local allocation) and the online-greedy algorithm (that picks an ordering over vertices in each batch and greedily allocates), as well the optimum offline. We observe that the competitive ratio of our algorithm beats the other algorithms in our simulations.

Experimental setup. We generate the following configuration allocation problem instances. There are 20 offline advertisers (i.e., $V \triangleq [20]$) and 100 arriving users (i.e., $U \triangleq [100]$).[21] Each advertiser j demands $n_j \triangleq 5$ number of impressions in total. For each user $i \in U$ and advertiser $j \in V$, we denote v_{ij} by the willingness-to-pay of advertiser j for an impression from user i (see the paragraph below about how we generate $\{v_{ij}\}$). For each user $i \in U$, the platform needs to choose an advertising configuration c which specifies a subset of advertisers $V_c \subseteq V$ to participate in a second-price auction for the arriving impression from user i: the winning advertiser $j^* = \arg\max_{j \in V_c} v_{ij}$ with the highest willingness-to-pay receives the impression from user i, and pays the per-impression price equal to the second highest willingness-to-pay among participating advertisers, i.e., $\xi_{i,c,j} \triangleq \mathbb{1}\{j = j^*\}$ and $w_{i,c,j} \triangleq v_{ij^\dagger}$ where $j^\dagger = \arg\max_{j \in V_c \setminus \{j^*\}} v_{ij}$.

We generate the willingness-to-pay quantities $\{v_{ij}\}$ as follows. First, we draw 20 ($\triangleq T$) i.i.d. samples from the exponential distribution, and denote them as $\hat{w}_1 \le \cdots \le \hat{w}_T$. Next, we draw 20 i.i.d. samples between $[1, 2]$ uniformly at random, and denote them as $\mu_1 \le \cdots \le \mu_{20}$. For each user $i \in U$ and advertiser $j \in V$, we set the willingness-to-pay $v_{ij} \triangleq \hat{w}_\tau$ where $\tau = \lceil \tau_i + \tau_{ij} \cdot \mu_j \rceil$ and both τ_i, τ_{ij} are drawn i.i.d. from $[0, \frac{T}{3}]$ uniformly at random. Note that from the ex ante perspective, the willingness-to-pay $\{v_{ij}\}$'s are symmetric over all users for each fixed advertiser j, and increasing (i.e., stochastic dominant) with respect to advertiser j (since μ_j is increasing in j) for each fixed user i.

We consider both scenarios where arriving users are ex ante symmetric and ex ante asymmetric, respectively. In the symmetric-user (resp. asymmetric-user) scenario, for each user $i \in U$, we randomly generate 5 feasible configurations, each of which corresponds to a second-price auction with 3 advertisers drawn uniformly at random from V (resp. $V \setminus [\lfloor \frac{17 \cdot i}{100} \rfloor]$). We assume that there are K stages, where each stage $k \in [K]$ contains a batch of users $U_k \triangleq [(k-1) \cdot \lceil \frac{100}{K} \rceil + 1 : \min\{100, k \cdot \lceil \frac{100}{K} \rceil\}]$. Note that in the asymmetric-user scenario, the advertisers who can participate in the auction for user i become more restrictive as i increases. In this way, policies have incentives to carefully preserve enough demands for later users.

Policies. In our numerical experiments, we compare the expected revenue of four different policies/benchmark:

1. *Online greedy policy* (`Online-GR`): greedily decide the fractional allocation for each user $i = 1, 2, \ldots, 100$. for each user $i = 1, 2, \ldots, 100$, given the current allocation, `Online-GR` greedily finds the fractional allocation which maximizes the immediate revenue improvement for user i.
2. *Batched greedy policy* (`Batched-GR`): for each stage $k \in [K]$, given the current allocation, `Batched-GR` greedily finds the fractional allocation which maximizes the immediate revenue improvement for users in batch U_k.

[21] We also ran our experiments by varying all parameters in our synthetic instances. We obtained similar results and verified the robustness of our numerical findings.

Table 2 The average competitive ratios for different policies.

	user-symmetric		user-asymmetric	
	$K=2$	$K=5$	$K=2$	$K=5$
`Online-GR`	0.913	0.913	0.849	0.849
`Batched-GR`	0.947	0.919	0.877	0.860
`PR-MCA`	0.982	0.961	0.937	0.903

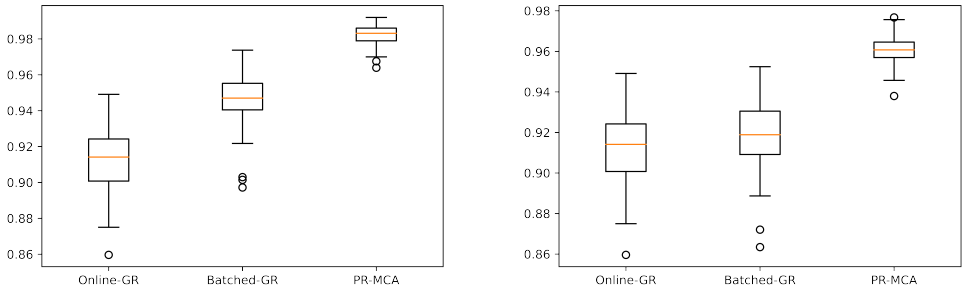
3. *Polynomial regularized max configuration allocation* (`PR-MCA`): this policy is Algorithm 2 designed in Section 4. It attains the optimal competitive ratio $\Gamma(K)$. We numerically solve convex program $\mathcal{P}_k^{\text{MCA}}$ in `PR-MCA` with the Frank–Wolfe algorithm (100 iterations per convex program, Frank and Wolfe, 1956).
4. *Optimum offline solution*: this benchmark is the solution of linear program PRIMAL-MCA in Section 2.2. It upperbounds the expected revenue of any multi-stage allocation algorithms.

Results. First, we randomly generate 100 configuration allocation problem instances for both user-symmetric and user-asymmetric scenarios with $K=2$ and $K=5$ stages. For each instance, we simulate all three policies defined above, and compute the ratio between the revenue of each policy against the revenue of the optimum offline solution. We summarize the competitive ratios for each policy in Table 2 and Figure 5. It can be observed that both `Batched-GR` and `PR-MCA` which utilize the batch arrival outperform `Online-GR`, and the ratio between better from $K=5$ batches to $K=2$ batches. Furthermore, `PR-MCA` which hedges the batch-wise allocation outperforms `Batched-GR` which greedily decides the batch-wise allocation. Specifically, by switching from `Online-GR` (resp. `Batched-GR`) to `PR-MCA`, there are strictly more than 5% revenue improvement in all scenarios.

To illustrate the the benefit of batching, we also plots the average (over 100 configuration allocation problem instances) of the ratios for `PR-MCA` and `Batched-GR` for $K=1,\ldots,10$ for both user-symmetric and user-asymmetric scenario in Figure 6. It can be observed that the average ratio is decreasing as the total number of stages K increases, and `PR-MCA` outperforms `Batched-GR` in both scenarios over all K.

E. Future Directions and Open Problems

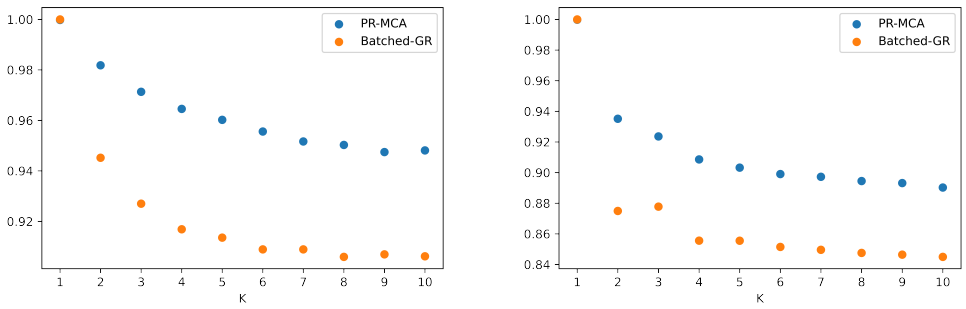
Several open questions arise from this work. The first question is to identify the optimal multi-stage algorithm and its competitive ratio for the multi-stage *integral* bipartite matching problem (i.e., the multi-stage version of the classic problem studied in Karp et al., 1990). Showing whether our competitive ratio can be obtained in the integral matching problem for $K>2$ is still an open problem. Second, inspired by the importance of batched allocations in applications such as ride-sharing and cloud computing, one might want to study the multi-stage resource allocation where the resources are *reusable*, similar to some recent work on online allocation of reusable resources (Feng et al., 2019, 2021; Goyal et al., 2020; Gong et al., 2022). In such a setting, a resource after being assigned to a demand request will be available again for reassignment after a certain time durations, which can be deterministic (and fixed over time) or i.i.d. over time. Note that in such a problem formulation, resource usage durations are described in the number of stages it takes for a resource to be available again. Third, our multi-stage matching algorithms assume knowing the number of stages K in advance. One can now investigate the question of having access to a predictor for K and asks the question of obtaining robustness and consistency bounds (in the same style as in the learning-augmented online algorithm literature). Fourth, one can consider applying



(a) user-symmetric, $K = 2$ (b) user-symmetric, $K = 5$

(c) user-asymmetric, $K = 2$ (d) user-asymmetric, $K = 5$

Figure 5 **Box and whisker comparison of different policies in terms of the competitive ratios. Results are based on 100 iterations of Monte-Carlo simulation.**



(a) user-symmetric (b) user-asymmetric

Figure 6 **The average competitive ratios for `PR-MCA` and `Batched-GR` as functions of the total number of stages K.**

our developed machinery and framework in this paper to multi-stage stochastic matching (Manshadi et al., 2012; Jaillet and Lu, 2014), or multi-stage matching with stochastic rewards (Mehta and Panigrahi, 2012). We conjecture that our approach can be extended to all of these settings and help with improved competitive ratios. Fifth, for the general model of the multi-stage configuration allocation, the proposed algorithm (as well as other algorithms in literature) requires optimization over the space of configurations. It would be interesting to identify combinatorial structures for these applications and design time-efficient subroutines for the aforementioned optimization over configurations. We leave further investigation around these directions and the possibility of extending our approach as open problems.

Finally, it is also worth studying the benefit of batching in environments with additional structure. Note that the optimal competitive ratio $\Gamma(K)$ converges to $1-\frac{1}{e}$ at a linear rate as K converges to infinity, i.e., $\Gamma(K)=1-\frac{1}{e}+O(\frac{1}{K})$. In other words, the benefit of batching decays quite quickly in the worst-case analysis. However, this fast decay might be an artifact of the model and the fact that designer aims to have a robust multi-stage allocation algorithms that perform well under any problem instance. To resolve this mystery, one can consider more refined models with additional (non-)stochastic structures, and identify algorithms that benefit from batching, but this time with the goal of highlighting the benefit of batching in a regime where the number of batches is super constant, but not very large. In such a model, another possible variable that can play a role is the average size of a batch. We leave all of these investigations and finding alternative problem formulations that can show the benefits of batching in non-stationary environments as an intriguing direction for future research.

F. Further Discussions on Practical Relevance and Applications

We discuss the computation of the proposed algorithm and the choice of time horizon (i.e., total number of stages K) in the following two subsections, respectively.

F.1. Solving Convex Program $\mathcal{P}_k^{\text{MCA}}$ in Practice

In the proposed algorithm `PR-MCA` (Algorithm 2), it solves the optimal solution of price-level regularized convex program $\mathcal{P}_k^{\text{MCA}}$ in each stage. It is worth noting that these types of convex program (especially program $\mathcal{P}_k^{\text{VWM}}$ pertaining to our base model of vertex-weighted matching) are generally solvable quickly for large-scale problems by using fast first-order methods such as projected gradient ascent (and generally the family of mirror ascent algorithm, e.g., see Bubeck et al., 2015), or conditional gradient ascent algorithm (e.g., Frank-Wolfe in Frank et al., 1956). We use Frank-Wolfe in our numerical simulations in Appendix D and could solve large-instances of our problem to an acceptable degree in a matter of seconds on a normal laptop. In fact, we posit that most of the time we only need a few iterations of a first-order method to find an acceptable solution for our purpose — and this is indeed what we have observed in our own numerical simulations on synthetic data (Appendix D). In applications such as video display ads, the allowance for latency is typically in the order of 150-600 seconds, which should give enough time for several iterations of a first-order method.

For the general version of multi-stage configuration allocation, there are complications in using first-order methods to solve the convex program quickly. In particular, we need a fast algorithm for solving linear optimization over the space of configurations (to be able to run an algorithm such as Frank-Wolfe). When the number of possible decorations/templates/configurations is not large (maybe a small constant), this task can be done quickly. It is worth noting that this complication exists in the online version of the problem, in particular in the problem of whole-page optimization (Feldman et al., 2009a; Devanur et al., 2016), which is a typical problem in display ads when the ad slate has combinatorial structures. In these settings, as stated in the aforementioned literature, typically the number of possible configurations for the ad-slate is small, which makes it amenable to search over the space of configurations.

F.2. Choice of Time Horizon K in Practice

In the base model of unweighted matching, the proposed Algorithm 1 does not require the knowledge of K and can use any strictly convex function as its regularizer (Remark 1). The polynomial regularizer depending on K are only used in its competitive ratio analysis. For more general models, if the algorithm does not know K but

knows an upper-bound $\bar{K}$ on K, $\Gamma(\bar{K})$ competitive ratio is immediately obtained by using the upper-bound in place of the true value of K.

Comparing the short-horizon and long-horizon scenarios, the ideal practical situation for running the proposed algorithm is a short-horizon problem with larger batches (and hence smaller K). In several display ads applications, including video ads, although the decision-making horizon is typically long, the proposed algorithm can be implemented as follows: divide the horizon into smaller rolling horizons, which we call them chunks, each with K stages for a parameter K chosen by the algorithm. Then we can run the proposed algorithm in each chunk separately (which requires splitting the total budget across these chunks first, probably equally or based on historical data, and with the possibility of roll-over from earlier chunks). With this view, the proposed algorithm proposes a better way to have competitive performance with respect to a K-lookahead algorithm, where the K-lookahead algorithm gets to see the entire arrival in each chunk in advance. Increasing K will essentially make this K look-ahead benchmark better estimate the offline optimum for the long-horizon problem and, at the same time, will decrease the competitive performance of the proposed algorithm against this benchmark (as $\Gamma(K)$ decreases as K increases). Whether the gain from batching can outweigh the loss due to more myopic outlook depends on the amount of non-stationarity among these chunks. If the environment is highly nonstationary, then it makes sense to only hope for local performance guarantee for each chunk versus a global performance guarantee against the offline optimum for long horizon, which can be achieved by a smaller choice of parameter K. In fact, considering a large K in such a case, say large enough to include the entire long horizon, although it will result in a constant competitive algorithm in theory, it is quite impractical, as the algorithm essentially hedges against a very distant point in future. In summary, the parameter K should be appropriately tuned based on whether we care about local performance versus global performance.

There are also scenarios in display ads where the horizon is effectively short. One such scenario is when there are demand shocks with a predicted end point and a lower bound on length (we set K so that the shock consists of K batches). For example, in the context of video ads, consider a scenario in which a football match is streamed globally at some point during the day. More concretely, consider the following infinite-horizon scenario: we keep running the convex programming algorithm with exponential function, which guarantees a competitive ratio of $1-1/e$ regardless (Remark 2). Now assume that the demand shock mentioned above occurs, and we estimate that it will last for K batches. The algorithm can now switch its regularizer to a sequence of polynomials with decreasing degrees and gradually decrease the hedging throughout the shock. If the demand shock is considerable, the algorithm should be able to get a better approximation versus the algorithm that uses the same regularizer.