New Approximations for Broadcast Scheduling
via Variants of α-point Rounding

Sungjin Im[*]   Maxim Sviridenko[†]

Abstract

We revisit the pull-based broadcast scheduling model. In this model, there are n unit-sized pages of information available at the server. Clients send their requests to the server over time asking for specific pages. The server can transmit only one page at each time. When the server transmits a page, all outstanding requests for the page are simultaneously satisfied, and this is what distinguishes broadcast scheduling from the standard scheduling setting where each job must be processed separately by the server. Broadcast scheduling has received a considerable amount of attention due to the algorithmic challenges that it gives in addition to its applications in multicast systems and wireless and LAN networks. In this paper, we give the following new approximation results for two popular objectives:

- For the objective of minimizing the maximum flow time, we give the *first* PTAS. Previously, it was known that the algorithm First-In-First-Out (FIFO) is a 2-approximation, and it is tight [14, 16]. It has been suggested as an open problem to obtain a better approximation [14, 4, 25, 31].
- For the objective of maximizing the throughput, we give a 0.7759-approximation which improves upon the previous best known 0.75-approximation [23].

Our key techniques for these improvements are novel variants of α-point rounding that can effectively reduce congestion in schedule which is often the main hurdle in designing scheduling algorithms based on linear programming. We believe that our new rounding schemes could be of potential use for other scheduling problems.

1 Introduction

We consider the pull-based broadcast scheduling model. In this model, there is a single server which contains n pages of useful information. Each client sends a request ρ asking for a specific page p. When the server broadcasts a page p, all outstanding/unsatisfied requests for the same page p are satisfied simultaneously. This is the main difference from the standard scheduling setting where each request needs to be processed separately by the server. This model is called "pull-based" since clients initiate the requests, while in the push-based model the server transmits pages according to the given frequency of pages requested.

Broadcast scheduling has applications in multicast systems, LAN and wireless systems [36, 5, 6]. We note that data broadcast scheduling is used in commercial systems [3, 2, 1], and it helps increase the system's bandwidth by serving multiple requests simultaneously. Also it can be viewed as a special case of batch scheduling which has been extensively studied in the stochastic and queueing theory literature [20, 19, 34, 35]. Broadcast scheduling has received a substantial amount of attention from the algorithms community, and has been studied both for the pull-based and push-based models, also both in the online and offline settings [11, 6, 5, 12]. This is because in addition to the aforementioned applications it gives algorithmic challenges concerning how to group requests for the same page over time to satisfy more requests with less transmissions while optimizing/satisfying certain scheduling objectives.

In this paper, we consider two objectives of minimizing the maximum flow time and maximizing the total throughput (profit). We first discuss the first objective. Each request ρ is released at time r_ρ asking for a specific page p_ρ. We assume that all pages are unit-sized, and requests arrive only at integer times. This unit-sized page assumption has been adopted in most previous literature. This assumption is justified when all pages have similar sizes, and still keeps the main difficulty of the problem. Consider any feasible schedule σ where at most one page is transmitted at each integer time. The completion time C_ρ^σ of request ρ is defined as the first time greater than r_ρ when page p_ρ is transmitted. If no such transmission exists, $C_\rho^\sigma = \infty$. Note that all requests have a flow time of at least one. The goal is to find a schedule σ that minimizes $\max_\rho (C_\rho^\sigma - r_\rho)$. If the schedule σ is clear from the context, it may be omitted.

This problem was first suggested in [12], which was

the paper with [28] that initiated the study of pull-based broadcast scheduling in the worst case analysis model. In fact, [12] claimed that the online algorithm First-In-First-Out (FIFO) is 2-competitive for this problem. However, it was fairly later that the formal proof was found [14, 16]. This problem was shown to be NP-hard [14]. Although the simple algorithm FIFO achieves 2-competitiveness, it has been the best known approximation guarantee even for the offline setting. Hence a natural open question was if one can obtain a better approximation in the offline setting [14, 25, 31]. Furthermore, this problem was mentioned in the Dagstuhl seminar on scheduling in 2010 as an open problem with an interesting connection to the so-called IRS Tax Scheduling problem [4].

In the other problem of maximizing the total throughput (profit), each request ρ is also associated with deadline d_ρ and profit w_ρ. If page p is transmitted during $[r_\rho + 1, d_\rho]$, the request yields profit w_ρ. This objective is also NP-hard to optimize [14]. There are several constant factor approximations known. The simple greedy (online) algorithm that transmits page p that satisfies the requests of the maximum total profit is known to be 2-competitive [29]. Other approaches are based on linear programming and rounding. As a high-level overview, the LP gives a fractional schedule $\{x_{p,t}\}$ over all pages p and time steps t such that $\sum_p x_{p,t} = 1$ for all time steps t. Here $x_{p,t}$ is the (possibly fractional) amount of page p that is transmitted at time t. The independent rounding of picking one page at each time t according to $x_{p,t}$ gives a $(1 - 1/e)$-approximation [23, 15]. The best approximation currently known uses the elegant dependent rounding in [23] which gives a rounding scheme for a bipartite graph while satisfying some hard constraints (here the relation between pages and times is described as a bipartite graph). The current best approximation factor is 0.75 [23].

Our Contributions and Techniques. One of our main results is the *first* polynomial time approximation scheme (PTAS) for the maximum flow time objective.

THEOREM 1.1. *There exists a PTAS for minimizing the maximum flow time in broadcast scheduling. More precisely, for any $0 < \epsilon \le 1$, there exists a $(1 + \epsilon)$-approximation algorithm with running time $m^{O(1/\epsilon^4)}$ where m is the number of requests.*

One of the key algorithmic ideas in obtaining this result lies in our novel group-based α-point rounding. The α-point rounding has been useful in rounding fractional solutions for scheduling problems. For examples and pointers, see [9, 24, 33]. As mentioned before, the LP relaxation will give the amount $x_{p,t}$ by which page p needs to be transmitted at time t. In the standard α-point rounding, for each page p, one random value α_p is picked uniformly at random from $[0, 1]$, and page p is transmitted at times t such that $\sum_{t'=1}^{t} x_{p,t'} \ge \alpha_p + k > \sum_{t'=1}^{t-1} x_{p,t'}$

for an integer k. The resulting (possibly infeasible) schedule has nice properties such as preserving the flow time of each request in expectation. However, it could result in a large congestion during some time interval I. Namely, too many transmissions may be made during I compared to $|I|$, the length of I, or equivalently the maximum number of transmissions that can be made during I. This overflow could be as large as the standard deviation $\Theta(\sqrt{n})$ for some interval; recall that n is the number of pages. To make this schedule feasible, transmissions are delayed by the amount of the overflow. In fact, this is why [9] solved a sequence of relaxed linear programs iteratively to avoid this large delay for the average flow objective. However, the upper bound on the overflow shown in [9] is $O(\log^2 n / \log \log n)$. Recently, the approximation guarantee was improved in [7] to $\tilde{O}(\log^{1.5} n)$ by combining the iterative rounding framework in [9] and a novel rounding scheme based on discrepancy theory. However, this approximation guarantee is still too large for our goal of obtaining a PTAS. Furthermore, in general, the overflow during any interval could have a more serious effect on our objective of minimizing the maximum flow time, while for other objectives such as the total flow time, the increase of flow time of some requests may be charged to other requests.

Our key idea is to partition pages into a small number of groups and to let all pages in the same group g to share a single random value $\alpha'_g \in [0, 1]$. Roughly speaking, the requests for any two different pages in the same group have substantially different release times. If the difference is more than L^*, the maximum flow time of an optimal solution, it can be assumed that two different pages in the same group are never transmitted at the same time, and this is precisely why the pages in the same group can share the same random variable. Although α_p for pages p in the same group g are completely determined by a single random value α'_g, the quantities α_p are not necessarily the same. To make the number of transmission made for pages in the same group g as close to the total (fractional) amount of transmission made for the pages as possible, we transmit a page for each group g at times t when the cumulative quantity $\sum_{p \in g} \sum_{t'=1}^{t} x_{p,t'}$ first exceeds $\alpha'_g + k$ for some integer k. To our best knowledge, this seemingly simple idea has never been used before, and we believe that it is worth further investigation for the potential use for other problems. By applying concentration inequalities with this small number of random variables (at most $O(L^*)$), we are able to show that the overflow is only $O(\epsilon L^*)$ for any interval if L^* is considerably big. We derandomize this process using the method of pessimistic estimators [32]. When L^* is small, we design a dynamic programming which completes our PTAS.

REMARK 1.1. *The reader may wonder if the algorithm*

FIFO can be strengthened by an LP. We however show that a natural LP-guided FIFO achieves only a 2-approximation. See Section 2.5.

The other main result of this paper is an improved approximation for the maximum throughput objective.

THEOREM 1.2. *For some $\alpha_0 > \frac{1}{2} + \frac{3}{4e}$ and any $\epsilon > 0$, there exists a $(\alpha_0 - \epsilon)$-approximation for maximizing the throughput (total profit) in broadcast scheduling $(\frac{1}{2} + \frac{3}{4e} > 0.7759)$. Furthermore the running time of the algorithm is in polynomial in $(1/\epsilon)^{O(1/\epsilon)}$ and m.*

REMARK 1.2. *We note that the 5/6-approximation claimed in [8] without proof was unfortunately wrong and excluded from the journal version [9].*

Our rounding algorithm for the throughput objective is very different from the current best approximation in [23] and other known approximations [29, 15]. Let us call the interval $[r_\rho + 1, d_\rho]$ request ρ's window. We classify requests into two groups, depending on their window sizes. Our algorithm has two main components. If small-window requests give a relatively large profit, we use a configuration LP to collect most profits from small window requests. Here by a configuration, we mean all possible transmissions that can be made during a small interval. The time horizon is partitioned into short disjoint intervals and configurations are defined for each of such intervals. The rounding is simply picking a configuration for each disjoint interval. Since configurations capture enough details for small-window requests, we will be able to collect most profits from those, while we achieve an $(1 - 1/e)$-approximation for large-window requests.

The other component of our algorithm is used when large-window requests give a large profit. We modify the α-point rounding in an interesting way. After the α-point rounding, if multiple transmissions are made at a time, we keep only one transmission at random using the fair contention resolution scheme in [22] (interestingly, this already achieves an $(1 - 1/e)$ approximation), and let other transmissions walk either to the right or to the left at random for a certain constant number of time steps to find an available empty time slot. More precisely, consider a transmission of page p at time t that is about to move to the left or to the right. Consider a large-window request ρ such that $t \in [r_\rho + 1, d_\rho]$ and $p_\rho = p$. Suppose that t is fairly far from d_ρ. Then if the transmission moves to the right and can find an empty time slot soon, specifically by d_ρ, then the transmission will still satisfy the request ρ, and this is how we get more profits from large-window requests. It now remains to guarantee that there are enough empty time slots available so that the random walk of transmissions can find new places with a probability arbitrarily close to 1. To this end, after the α-point rounding, we initially free away an ϵ-fraction of time

slots, whose effect will be negligible to the approximation factor.

Although we use configurations that encode all possible transmission for an interval of length depending on ϵ, note that the running time is in polynomial in $(1/\epsilon)^{O(1/\epsilon)}$ and the number of requests. This is achieved by solving the dual of the LP using an efficient separation oracle.

The flow time and throughput objectives are perhaps the most popular ones in the scheduling literature. To optimize flow times, it is crucial to minimize congestion during any interval since we are required to satisfy all requests, and congestion can accumulate over time, thereby increasing overall flow time considerably. On the other hand in the throughput objective the time constraints are hard but we are allowed to discard some requests. We believe that our variants of α-point rounding that reduce congestion and resolve conflicts could be of potential use for other scheduling problems as well.

Related Work. A submodular generalization of the maximum throughput objective was studied in [15]. In this extension, each request ρ is associated with a submodular profit function that is defined over the times when page p_ρ is transmitted. For this problem, [15] gave a 0.5-competitive algorithm and a $(1 - 1/e)$-approximation algorithm. Other variants of this problem were considered and constant competitive algorithms were given [37, 13, 18]. [23] gives a 0.75-approximation for the maximum throughput objective in a slightly more general setting.

As mentioned before, the maximum flow time objective was first considered in [12]. Chang et al. gave the first proof for the claim that FIFO is 2-competitive for the objective [14]. Later, Chekuri et al. extended the result to varying-sized pages using a different proof [16]. The performance guarantee of FIFO is tight [14], and it remains the case even if randomization is used to break ties between pages [15]. In the 2010 Dagstuhl seminar on scheduling, a special case for the throughput objective was introduced under the name of the IRS Tax Scheduling problem [4]. The problem, explained in broadcast scheduling terminology, asks if there is a feasible schedule that satisfies all requests ρ during their window $[r_\rho + 1, d_\rho]$ when there are at most two requests for each page. It was shown that an exact polynomial time algorithm for the Tax Scheduling problem can be used to obtain a randomized algorithm that achieves a 2-approximation for the maximum flow time while guaranteeing each request's expected flow time is at most 1.75 times the maximum flow time of the optimal solution [17]. It remains open if the Tax Scheduling problem is NP-hard or not.

Other interesting objectives were studied in broadcast scheduling. For the average flow time objective, [9] gave an $O(\log^2 n / \log \log n)$-approximation. We note

that the algorithm and analysis in [9] can be used to give the same approximation guarantee for minimizing the maximum weighted flow time. As mentioned before, this approximation guarantee was recently improved in [7] to $\tilde{O}(\log^{1.5} n)$. In the online setting, there exists a $\Omega(n)$ lower bound on the competitive ratio for the average flow time objective, hence to overcome this lower bound, a relaxation called speed augmentation was used [27]. In this relaxation, the online algorithm runs on a machine that is $(1 + \epsilon)$ times faster than the machine the optimal offline scheduler runs on. An online algorithm is said to be scalable if it is $f(\epsilon)$-competitive for any $\epsilon > 0$. For average flow time and its variants, scalable algorithms were given [26, 10, 21]. There is a scalable algorithm known for the maximum weighted flow time objective [16].

Formal Problem Definition and Notation. There are n pages of information, and all pages are unit-sized. We will denote page mostly as p, q, and denote the set of pages as $\mathcal{P}$. Times are slotted. Requests can arrive only at a non-negative integer time. The server can transmit at most one page at each positive integer time. A request ρ is released at time r_ρ asking for a specific page p_ρ. We let $\mathcal{R}$ denote the entire set of requests. We may add some condition as a subscript to denote a subset of requests that satisfy the condition. For example, $\mathcal{R}_{r \geq t}$ will refer to the requests that arrive no earlier than time t. Let $m := |\mathcal{R}|$ denote the number of requests. Without loss of generality, we assume that $m \geq n$. Also the set of times when page p is requested is denoted as $\mathcal{T}_p$. We say that a schedule σ is feasible if at most one page is transmitted at each time. In a schedule σ, request ρ is completed at the first time $t' > r_\rho$ when the page p_ρ is transmitted. The flow time $F_\rho^\sigma := C_\rho^\sigma - r_\rho$ of a request is the length of time that ρ waits since its arrival until its completion. In the problem of minimizing the maximum flow time, the goal is to find a feasible schedule that minimizes $\max_{\rho \in \mathcal{R}} F_\rho^\sigma$. As mentioned before, if the schedule σ is clear from the context, it may be omitted. In the problem of maximizing the throughput (profit), each request ρ has weight w_ρ and deadline d_ρ. If page p_ρ is transmitted during its window $W_\rho := [r_\rho + 1, d_\rho]$, then we get a profit of w_ρ. The objective is to obtain a feasible schedule that gives the maximum total profit.

We let T denote the maximum time we need to consider, which is at most $\max_{\rho \in \mathcal{R}} r_\rho + n$. This is because transmitting all n pages after the release of the last request completes all requests, and there is no incentive of transmitting the same page more than once when there are no more requests to arrive. We assume that all requests are given explicitly, and hence it follows that the input size is as large as m, the number of requests. We suggest the reader to read Section 2 and 3 assuming that T is $O(m)$. Intuitively, if requests rarely arrive, at a rate of less than

one request per time on average, the problem becomes easier. Later we will show how to remove this assumption.

2 A PTAS for the Maximum Flow Time

In this section, we give a PTAS for the problem of minimizing the maximum flow time in broadcast scheduling. Consider any instance J of requests. Let L^* denote the maximum flow time of an optimal schedule on the instance. Note that $1 \leq L^* \leq n$, since we can satisfy all requests within n time steps by repeatedly transmitting all n pages. We can without loss of generality assume that the quantity L^* is known to the algorithm; this can be easily done by performing a binary search on the value of L^* in the range of $[1, n]$. To make our algorithm and analysis more transparent, we will throughout assume that $T = O(m)$. In Section 2.3, this simplifying assumption will be removed. We will proceed with our analysis by considering two cases depending on the value of L^*, and show the following in Section 2.1 and 2.2, respectively.

LEMMA 2.1. *Suppose that $L^* \geq \frac{1}{\epsilon^3} \log T$. Then there exists a randomized algorithm that yields a feasible schedule with the maximum flow time of at most $(1+6\epsilon)L^*$ with a probability of at least $1 - 1/T$.*

THEOREM 2.1. *For any $0 < \epsilon \leq 1$, there exists a $(1 + 6\epsilon)$-approximation for the maximum flow time in broadcast scheduling with run time $O\left(\frac{T}{\epsilon L^*} \cdot (L^*)^2\right) \cdot \left(\frac{3}{\epsilon}\right)^{18L^*} \cdot \left(\frac{6e}{\epsilon}\right)^{\epsilon L^*}$.*

Lemma 2.1 and Theorem 2.1 would easily yield a randomized version of one of our main results.

THEOREM 2.2. *There exists a randomized PTAS for minimizing the maximum flow time in broadcast scheduling. More precisely, there exists a randomized $(1 + \epsilon)$-approximation with run time $m^2 \cdot T^{O(1/\epsilon^4)}$ that succeeds with a high probability.*

Proof. The run time is upper-bounded by the quantity $O\left(T \cdot (L^*)^2\right) \cdot \left(\frac{3}{\epsilon}\right)^{18L^*} \cdot \left(\frac{6e}{\epsilon}\right)^{\epsilon L^*}$ with $L^* = (1/\epsilon^3) \log T$, which simplifies to

$$O(T \cdot m^2) \cdot (3/\epsilon)^{(18/\epsilon^3) \log T} \cdot (6e/\epsilon)^{(1/\epsilon^2) \log T}$$
$$\leq O(T \cdot m^2) \cdot (3/\epsilon)^{(18/\epsilon^3) \log T} \cdot (3/\epsilon^3)^{(3/\epsilon^2) \log T}$$
$$\leq O(T \cdot m^2) \cdot (3/\epsilon)^{(21/\epsilon^3) \log T}$$
$$= O(T \cdot m^2) \cdot T^{\frac{21}{\epsilon^3} \log \frac{3}{\epsilon}}$$
$$= m^2 \cdot T^{O(\frac{1}{\epsilon^4})}$$

$\square$

As mentioned before, we will remove the dependency of run time on T in Section 2.3. Also in Section 2.4, using the method of pessimistic estimators, we will derandomize the rounding scheme described in Section 2.1. This

will make the result stated in Theorem 2.2 deterministic, thereby completing the proof of Theorem 1.1. We note that one can improve the run time of our randomized algorithm to $m^2 \cdot T^{O(1/\epsilon^3)}$ by considering two cases that $L^* \geq (1/\epsilon^2) \log T$ or not. However, the corresponding derandomization process seems more tricky and involved, hence we present the analysis of the result with a slightly worse run time.

We begin with the case when L^* is large.

2.1 Case: $L^* \geq (1/\epsilon^3) \log T$.
This section is devoted to proving Lemma 2.1. For notational convenience, we begin with simplifying the given instance of requests. We say that two requests ρ and ρ' for the same page p are adjacent if there are no requests for page p released between time r_ρ and $r_{\rho'}$. We claim that we can without loss of generality assume that any two adjacent requests for the same page arrive within less than L^* time steps. To see this, suppose that there are two adjacent requests ρ and ρ' for page p that arrive apart by at least L^* time steps. Then in the optimal schedule, no transmission of page p can be used to satisfy ρ and ρ' simultaneously. Hence we can assume that the requests in $\mathcal{T}_{p, r \leq r_\rho}$ and those in $\mathcal{T}_{p, r \geq r_{\rho'}}$ are for different pages, where $\mathcal{T}_{p, r \leq r_\rho}$ and $\mathcal{T}_{p, r \geq r_{\rho'}}$ denote the set of requests for page p that arrive no later than ρ and no earlier than ρ', respectively.

Henceforth, we assume that any two adjacent requests ρ, ρ' (for the same page) arrive within less than L^* time steps, i.e. $|r_\rho - r_{\rho'}| \leq L^* - 1$. We consider the following integer programming that determines if there is a feasible schedule with the maximum flow time of at most L^*.

($\mathsf{LP_{MaxFlow}}$)

(2.1)
$$\sum_{t'=t+1}^{t+L^*} x_{p,t'} \geq 1 \qquad \forall p \in \mathcal{P}, t \in \mathcal{T}_p$$

(2.2)
$$\sum_{p \in \mathcal{P}} x_{p,t} \leq 1 \qquad \forall t \in [T]$$

(2.3) $\quad x_{p,t} = 0 \ \ \forall p \in \mathcal{P}, t \notin [\min \mathcal{T}_p + 1, \max \mathcal{T}_p + L^*]$
$$x_{p,t} \in \{0,1\} \, \forall p \in \mathcal{P}, t \in [T]$$

The first constraints say that all requests must be completed within L^* time steps. The second constraints state that at most one page can be transmitted at each time. The third constraints ensure that no transmission is made if it cannot be used to satisfy a request within L^* time steps. We relax the integer programing by replacing the last constraints with $x_{p,t} \geq 0$. We solve $\mathsf{LP_{MaxFlow}}$ and let $x^*_{p,t}$ denote the optimal fractional solution. Our rounding

scheme will be based on the groups we define as follows. For each page p, define $W_p := [\min \mathcal{T}_p + 1, \max \mathcal{T}_p + L^*]$, which we call page p's window. Note that there exists an optimal schedule where page p is broadcast only during W_p. We now partition pages into groups $\mathcal{G}$ such that all pages in the same group $g \in \mathcal{G}$ have disjoint windows. We show that $2L^*$ groups suffice for this partition. The proof easily follows by observing that if there are too many different pages with similar release dates, then the optimal schedule must have a large value of the maximum flow time.

LEMMA 2.2. *We can in polynomial time partition pages $\mathcal{P}$ into at most $2L^*$ groups in $\mathcal{G}$ such that for any $p \neq q$ in the same group g, W_p and W_q are disjoint.*

Proof. We create an interval graph G with one interval corresponding to each window W_p. The partition of pages into groups corresponds to a feasible vertex coloring of G. It is well known that the chromatic number of an interval graph is equal to its clique number. Moreover, such a coloring can be easily found by a greedy algorithm.

We claim that a clique number of G is upper bounded by $2L^*$. To show this we prove that at any time t, there are at most $2L^*$ windows W_p that contain t. To this end, it suffices to show that if $t \in W_p$, then p must be transmitted at least once during $[t - L^* + 1, t + L^*]$. We consider two cases. If $t \geq \max \mathcal{T}_p$, then this claim holds since to satisfy the last request, page p must be transmitted during $[\max \mathcal{T}_p + 1, \max \mathcal{T}_p + L^*]$, which is contained in $[t - L^* + 1, t + L^*]$. Otherwise, there must exist a request for p that arrives at some time $t' \in [t - L^* + 1, t]$, and to satisfy the request, page p must be transmitted at least once during $[t - L^* + 2, t + L^*]$. $\qquad \square$

We now describe our rounding scheme. For each group $g \in \mathcal{G}$, we define a cumulative amount of transmission made by the fractional solution. Formally, define $y^*_{g,t} := \sum_{p \in g} \sum_{t' \leq t} x^*_{p,t'}$. Now for each group g, pick α_g from $[0, 1]$ uniformly at random, and this is the only random value for group g. We first obtain a tentative schedule σ_{temp} and then the final schedule σ_{final}. In the tentative schedule σ_{temp}, all requests are satisfied within L^* time steps, but it is allowed to transmit more than one page at a time. In σ_{temp}, for each group g, we transmit a page p in g at times t such that $y^*_{g,t-1} < k + \alpha_g \leq y^*_{g,t}$ for some integer k. Here the page $p \in g$ transmitted at time t for group g is such that $x^*_{p,t} > 0$. Note that for any time t, there is at most one page $p \in g$ with $x^*_{p,t} > 0$. This is due to the definition of groups and the constraints (2.3). We now transform σ_{temp} into the final feasible schedule σ_{final} in the First-In-First-Out fashion: Think of a transmission of page p at time τ as a job $j(p, \tau)$ that arrives at time τ, and add the job to the queue. At each time t, we dequeue the job $j(p, \tau)$ with the earliest arrival time τ

and transmit page p. Clearly, in the final schedule σ_{final}, at most one transmission is made at each time. We will show that in σ_{final}, all requests have flow time at most $(1 + 6\epsilon)L^*$ with high probability.

We begin our analysis with the following easy proposition concerning the tentative schedule σ_{temp}.

PROPOSITION 2.1. *In σ_{temp}, all requests have flow time at most L^*.*

Proof. Consider any request ρ for page p, and let $g \in \mathcal{G}$ be the group that contains page p. Then due to constraints (2.1), we have $y^*_{g,r_\rho+L^*} - y^*_{g,r_\rho} = \sum_{\tau=r_\rho+1}^{r_\rho+L^*} x^*_{p,t} \geq 1$. Hence for any random value α_g, a page is transmitted from g during $[r_\rho+1, r_\rho+L^*]$. Further, since $[r_\rho+1, r_\rho+L^*] \in W_p$ and all pages in g have disjoint windows, page p is transmitted during $[r_\rho + 1, r_\rho + L^*]$. $\square$

We complete the analysis by showing that w.h.p. each request's flow time does not increase too much in the transformation from σ_{temp} into σ_{final}. To this end, we need to measure the number of transmissions made in σ_{temp} during an interval I. Let $Q_{temp}(I)$ denote this quantity. We need the following simple lemma, which was shown in [9]. The lemma shows how the quantity $Q_{temp}(I)$ is related to the increase of each request's flow time from σ_{temp} to σ_{final}.

LEMMA 2.3. *For all requests ρ, $F^{\sigma_{final}}_\rho \leq F^{\sigma_{temp}}_\rho + \max_{1 \leq t_1 \leq t_2 \leq T} \max\{Q_{temp}([t_1, t_2]) - (t_2 - t_1 + 1), 0\}$.*

Proof. Consider any request ρ. Let t_0 denote the latest time $t < r_\rho$ such that there is only one transmission made at time t in σ_{temp} and the transmission is scheduled exactly at time t also in σ_{final} without any delay, or no transmission is made at time t in σ_{final}. If no such time t exists, then $t_0 = 0$. Note that $F^{\sigma_{final}}_\rho - F^{\sigma_{temp}}_\rho \leq (Q_{temp}([t_0 + 1, r_\rho]) - (r_\rho - (t_0 + 1) + 1)$. Hence the lemma follows. $\square$

Motivated by the above lemma, we define the overflow during $[t_1, t_2]$,

$OF([t_1, t_2]) := \max\{Q_{temp}([t_1, t_2]) - (t_2 - t_1 + 1), 0\}$.

Formally, we will upper-bound the overflow during an interval as follows. The proof uses the fact that there are $O(L^*)$ random variables that contribute to the overflow, and each random variable can contribute at most 1. This is where the page grouping idea plays a crucial role. The proof follows by using these facts and applying the Bernstein's concentration inequality. By a simple union bound over all possible intervals (at most T^2), the following lemma will imply Lemma 2.1.

LEMMA 2.4. *Suppose that $L^* \geq (1/\epsilon^3) \log T$. Then for any interval I, $\Pr[OF(I) \geq 6\epsilon L^*] \leq 1/T^3$.*

Proof. Consider any fixed interval $I = [t_1, t_2]$. For each group $g \in \mathcal{G}$, let N_g denote the number of transmissions that are made for pages in g during I in σ_{temp}, which is of course a random variable. Let $v_g := \sum_{p \in g} \sum_{t \in I} x^*_{p,t}$ denote the volume that $\mathsf{LP_{MaxFlow}}$ transmits for the pages in g during I. Note that $\mathbb{E}[N_g] = v_g$. Also observe that $X_g := N_g - \lfloor v_g \rfloor$ is a 0-1 random variable. This is because a page in g is transmitted in σ_{temp} at every time step $\mathsf{LP_{MaxFlow}}$ makes one additional volume of transmission for pages in g (except the first one which is done when α_g volume of transmission is done by $\mathsf{LP_{MaxFlow}}$). Note that X_g are independent with $\Pr[X_g = 1] = v_g - \lfloor v_g \rfloor$. To see this, let $v'_g := \sum_{p \in g} \sum_{t < t_1} x^*_{p,t}$. If $v'_g - \lfloor v'_g \rfloor + v_g - \lfloor v_g \rfloor \leq 1$, then $X_g = 1$ if $v'_g - \lfloor v'_g \rfloor \leq \alpha_g \leq v'_g - \lfloor v'_g \rfloor + v_g - \lfloor v_g \rfloor$, otherwise $X_g = 0$. If $v'_g - \lfloor v'_g \rfloor + v_g - \lfloor v_g \rfloor > 1$, then $X_g = 1$ if $v'_g - \lfloor v'_g \rfloor \leq \alpha_g \leq 1$ or $0 \leq \alpha_g \leq v'_g - \lfloor v'_g \rfloor + v_g - \lfloor v_g \rfloor - 1$, otherwise $X_g = 0$. By Lemma 2.2, we know that there are at most $2L^*$ random variables X_g. For notational convenience, let $\mu_g := \mathbb{E}[X_g]$ and $\mu := \sum_{g \in \mathcal{G}} \mu_g$.

We can relate the probability in the claim to the following probability of an event in terms of X_g.

$$\Pr\left[OF(I) \geq 6\epsilon L^*\right]$$
$$\leq \Pr\left[\sum_{g \in \mathcal{G}} N_g - \sum_{g \in \mathcal{G}} v_g \geq 6\epsilon L^*\right]$$
$$= \Pr\left[\sum_{g \in \mathcal{G}}(N_g - \lfloor v_g \rfloor) \geq \sum_{g \in \mathcal{G}}(v_g - \lfloor v_g \rfloor) + 6\epsilon L^*\right]$$
$$= \Pr\left[\sum_{g \in \mathcal{G}} X_g \geq \mu + 6\epsilon L^*\right]$$

The first inequality follows from the facts that $Q_{temp}(I) = \sum_{g \in \mathcal{G}} N_g$ and $\sum_{g \in \mathcal{G}} v_g = \sum_{g \in \mathcal{G}} \sum_{p \in g} \sum_{t \in I} x^*_{p,t} \leq t_2 - t_1 + 1$. The last equality comes from the definition of X_g. Knowing that $\mathtt{Var}[\sum_{g \in \mathcal{G}} X_g] \leq 2L^*$, by applying Theorem A.1 (Bernstein Concentration Inequality) with $\Delta = 6\epsilon L^*$, $V \leq 2L^*$ and $b \leq 1$, we derive

$$\Pr\left[\sum_{g \in \mathcal{G}} X_g \geq \mu + 6\epsilon L^*\right]$$
$$\leq \exp\left(-\frac{\Delta^2}{2V + (2/3)b\Delta}\right)$$
$$\leq \exp\left(-\frac{(6\epsilon L^*)^2}{4L^* + 4\epsilon L^*}\right) < \exp(-3\epsilon^2 L^*) \leq \frac{1}{T^3}$$

$\square$

2.2 Case: $L^* \leq (1/\epsilon^3) \log T$. We first observe that if L^* is a constant, the problem can be solved in polynomial time using dynamic programming. We first give the key idea of our dynamic programming and then describe it in detail. Suppose we need to decide which page to transmit at time t knowing the last L^* transmissions made

during $[t - L^*, t - 1]$ in the optimal schedule. Then in the remaining scheduling decision, we only need to care about $\mathcal{R}_{r \geq t-L^*}$, the requests that arrive no earlier than $t - L^*$. This is because all the other requests, $\mathcal{R}_{r < t-L^*}$ must be satisfied by time $t - 1$ in the optimal schedule. Since we do not know the "correct" last L^* transmissions, we simply enumerate all possible cases, whose number is at most n^{L^*}. Further, for each case, we keep track of if there exists a feasible schedule where all requests in $\mathcal{R}_{r < t-L^*}$ are finished within L^* time steps. Based on this observation, we propose the following dynamic programming.

Let L denote our guess of L^*; we will simply consider L in increasing order starting from 1. For $t \geq L - 1$, let $\mathcal{Q}(t) := \{\{(t - L + 1, p_{t-L+1}), (t - L + 2, p_{t-L+2}), ..., (t, p_t)\} \mid p_{t-L+1}, p_{t-L+2}, ..., p_t \in \mathcal{P}\}$ denote the collection of all possible transmissions that can be made during $[t - L + 1, t]$; here a pair $(t', p'_{t'})$ implies that page $p_{t'}$ is transmitted at time t'. Let us call an element $Q \in \mathcal{Q}(t)$ as a configuration with respect to time t. Note that $|\mathcal{Q}(t)| = n^L$. We say that a configuration $Q \in \mathcal{Q}(t)$ is *feasible* if there exists a schedule that is compatible with Q where all requests in $\mathcal{R}_{r \leq t-L}$ are completely satisfied by t and have flow time at most L.

We are now ready to describe our algorithm. Let $\mathcal{Q}^f(t)$ denote all feasible configurations in $\mathcal{Q}(t)$. We will compute $\mathcal{Q}^f(t)$ in increasing order of t starting from $t = L - 1$ to $t = T$. Note that $\mathcal{Q}^f(L - 1) = \mathcal{Q}(L - 1)$. To compute $\mathcal{Q}^f(t)$ from $\mathcal{Q}^f(t - 1)$, we consider each pair of a configuration $Q \in \mathcal{Q}^f(t - 1)$ and a page $p_t \in \mathcal{P}$. Let $Q = \{(t - L, p_{t-L}), (t - L + 1, p_{t-L+1}), ..., (t - 1, p_{t-1})\}$. We add $Q' = \{(t - L + 1, p_{t-L+1}), (t - L + 2, p_{t-L+2}), ..., (t, p_t)\}$ as a feasible configuration to $\mathcal{Q}^f(t)$ if all requests with release time $t - L$ are satisfied by $p_{t-L+1}, p_{t-L+2}, ..., p_{t-1}$ or p_t. Finally, if $\mathcal{Q}^f(T) = \emptyset$, then declare that there is no feasible schedule with the maximum flow time of at most L. Otherwise, it is straightforward to find a desired feasible schedule using the standard backtracking method, i.e. $\mathcal{Q}^f(T), \mathcal{Q}^f(T - 1), \mathcal{Q}^f(T - 2), ..., \mathcal{Q}^f(L - 1)$. For completeness, we present the pseudocode of this algorithm. See Algorithm 1 in Appendix B.

The correctness of the algorithm hinges on the correctness of $\mathcal{Q}^f(t)$, and it can be shown by a simple induction on t. Since the algorithm considers at most T time steps, and at each time t it considers all pairs of a configuration in $\mathcal{Q}^f(t - 1)$ and a page $p_t \in \mathcal{P}$, whose total number is at most $n^L \cdot n$, we derive the following lemma (recall that $m \geq n$).

LEMMA 2.5. *There exists an optimal algorithm with run time $O(T) \cdot m^{O(L^*)}$ for the problem of minimizing the maximum flow time in broadcast scheduling.*

Due to Lemma 2.5, we can assume that L^* is a sufficiently large constant. The dynamic programming

for general L^* is very similar to that for constant L^*. However, to obtain a slightly better running time which is needed to complete our PTAS, we need one more idea. The additional idea is that one can essentially assume that requests arrive only at integer multiples of ϵL^*, and transmissions can be made only at those time steps – now at most ϵL^* transmissions can be made at each of those time steps.

Let J denote the given instance of requests. We simplify J to make our analysis easier. After the modification, we will distinguish time steps only to the extent that is sufficient to allow a PTAS. The simplification is described as follows: Shift each request ρ's arrival time to the right to the closest integral multiple of ϵL^*, i.e. $\epsilon L^* \lceil \frac{r_\rho}{\epsilon L^*} \rceil$. Let J' denote this modified instance. We will assume that ϵL^* is an integer. This is justified by Lemma 2.5 which shows a polynomial time algorithm for any constant L^*. Assuming that L^* is a sufficiently large constant, we can find an approximate value L that is a multiple of $1/\epsilon$ and multiplicatively close to L^* in any arbitrary precision. The effect of this assumption on approximation factor and run time will be factored in later. Also we assume that $1/\epsilon$ is an integer. For the modified instance J', the scheduler is allowed to transmit only at times that are integral multiples of ϵL^*, and at each of such times, at most ϵL^* pages. The following lemma shows that this modification almost preserves the maximum flow time and that a good schedule for J' can be transformed into a good schedule for J. Recall that F^σ_ρ denotes ρ's flow time in schedule σ.

LEMMA 2.6. *There exists a schedule σ' for J' with the maximum flow time of at most $(1 + 2\epsilon)L^*$. Further, any schedule σ' for J' can be converted into a feasible schedule σ for the original instance J such that the flow time of each request increases by at most $2\epsilon L^*$, i.e. $F^\sigma_\rho - F^{\sigma'}_\rho \leq 2\epsilon L^*$ for all requests ρ.*

Proof. We first show the first claim. Observe that shifting all transmissions in the optimal schedule to the right, to the closest multiple of ϵL^* with additional ϵL^* time steps, is a feasible schedule for J'. Further this increases any request's flow time by at most $2\epsilon L^*$. Thus the first claim follows. We now turn our attention to the second claim. Recall that in σ', at times $\epsilon L^* k$ for some integer k, at most ϵL^* transmissions are made. Now move those transmissions to the times $\epsilon L^* k, \epsilon L^* k + 1, \epsilon L^* k + 2, ..., \epsilon L^* k + \epsilon L^* - 1$. This may increase the flow time of each request by at most ϵL^*. Since the arrival time of any request differs by at most ϵL^* between two instances, σ and σ', the second claim follows. $\square$

For notational simplicity, we will shrink the time horizon of the modified instance J' by a factor of ϵL^*. Note that now in J' requests can arrive at any positive

integer time, and at most ϵL^* transmissions can be made at each integer time. Hence the maximum flow time of the optimal schedule for J' is at most $(2+1/\epsilon)$ by Lemma 2.6. We slightly modify Algorithm 1 (for constant L^*) to give a polynomial time algorithm for the modified instance J' (for any fixed ϵ). Throughout, let $\ell := 2 + 1/\epsilon$. We first make a couple of useful observations. Let $\mathcal{P}_t$ denote the pages requested at time t, and $\mathcal{A}_t$ the pages transmitted by our algorithm at time t.

PROPOSITION 2.2. *There exists an optimal schedule $\mathcal{A}_t$ for J' with the maximum flow time ℓ such that*

- *for any page p and time t, at most one transmission is made for page p at time t.*

- *for any interval I of length ℓ and for any page p, there are at most two transmissions made for page p during I.*

- *for any time t, $\mathcal{A}_t \subseteq \bigcup_{i=1}^{\ell} \mathcal{P}_{t-i}$.*

Proof. Obviously there is no incentive of broadcasting a page more than once at the same time. Suppose there is an interval of length ℓ during which page p is transmitted more than twice. Then we can remove a middle transmission while keeping the maximum flow time no greater than ℓ. The final claim follows since if it were not true, any transmission of page p that was not requested for the last ℓ time steps would be wasted. $\quad\square$

PROPOSITION 2.3. *For any t, the number of pages that could be potentially transmitted at time t for any feasible schedule of the modified instance with maximum flow time ℓ is at most $|\bigcup_{i=1}^{\ell} \mathcal{P}_{t-i}| \leq 6L^*$. Further, the number of pages that could be potentially transmitted during $[t - \ell, t-1]$ for any feasible schedule of the modified instance with maximum flow time ℓ is at most $|\bigcup_{i=2}^{2\ell} \mathcal{P}_{t-i}| \leq 9L^*$.*

Proof. All pages in $\bigcup_{i=1}^{\ell} \mathcal{P}_{t-i}$ must be transmitted during $[t - \ell + 1, t + \ell - 1]$. Hence the total number of such pages is at most $2\ell * \epsilon L^* = 2\epsilon L^*(2 + 1/\epsilon) \leq 6L^*$. The last property in Proposition 2.2 completes the proof of the first claim. The second claim can be proven similarly. By the last property in Proposition 2.2, we have that $\bigcup_{i=1}^{\ell} \mathcal{A}_{t-i} \subseteq \bigcup_{i=2}^{2\ell} \mathcal{P}_{t-i}$. Since all pages in $\bigcup_{i=2}^{2\ell} \mathcal{P}_{t-i}$ must be transmitted during $[t - 2\ell + 1, t + \ell - 1]$, we conclude that $|\bigcup_{i=2}^{2\ell} \mathcal{P}_{t-i}| \leq 9L^*$. $\quad\square$

For any $t \geq \ell - 1$, we say that $\{(t - \ell + 1, \mathcal{A}_{t-\ell+1}), (t - \ell + 2, \mathcal{A}_{t-\ell+2}), ..., (t, \mathcal{A}_t)\}$ is a configuration with respect to time t if for all $0 \leq i < \ell, \mathcal{A}_{t-i} \subseteq \mathcal{P}$ and $|\mathcal{A}_{t-i}| \leq \epsilon L^*$. In other words, a configuration represents all transmissions made during $[t - \ell + 1, t]$. Here $\mathcal{A}_t$ can be any subset of at most ϵL^* pages requested in the last ℓ time steps. Let $\mathcal{Q}(t)$ denote the collection of all

configurations with respect to time t. Further we say that $Q \in \mathcal{Q}(t)$ is feasible if there exists a schedule compatible with Q where $\mathcal{R}_{r \leq t-\ell}$, all requests arriving no later than $t - \ell$ are completed by time t and have flow time at most ℓ. Let $\mathcal{Q}^f(t)$ denote the collection of all feasible configurations in $\mathcal{Q}(t)$.

We are now ready to describe our algorithm. This algorithm is very similar to Algorithm 1 for constant L^*. We compute $\mathcal{Q}^f(t)$ in increasing order of time t starting from ℓ to $T' := \lceil \frac{\max_\rho r_\rho}{\epsilon L^*} \rceil + \ell$. Note that $\mathcal{Q}^f(\ell - 1) = \mathcal{Q}(\ell - 1)$. To obtain $\mathcal{Q}^f(t)$ from $\mathcal{Q}^f(t-1)$, we consider each pair of $Q \in \mathcal{Q}^f(t-1)$ and a potential set of pages $\mathcal{A}_t$. Let $Q = \{(t - \ell, \mathcal{A}_{t-\ell}), (t - \ell + 1, \mathcal{A}_{t-\ell+1}), ..., (t - 1, \mathcal{A}_{t-1})\} \in \mathcal{Q}^f(t-1)$. Here $\mathcal{A}_t$ satisfy (1) all requests with release time $t - \ell$ are satisfied by pages in $\bigcup_{i=0}^{\ell-1} \mathcal{A}_{t-i}$, and (2) no page appears more than twice in $\mathcal{A}_{t-\ell+1}, \mathcal{A}_{t-\ell+2}, ..., \mathcal{A}_t$ (The second condition is justified by the second property in Proposition 2.2). If it is the case, then we add $Q' = \{(t - \ell + 1, \mathcal{A}_{t-\ell+1}), (t - \ell + 2, \mathcal{A}_{t-\ell+2}), ..., (t, \mathcal{A}_t)\}$ to $\mathcal{Q}^f(t)$. At the end of the algorithm, if $\mathcal{Q}^f(T') = \emptyset$, we declare that any feasible schedule has the maximum flow time larger than ℓ. Otherwise, using $\mathcal{Q}^f(\cdot)$, we construct a feasible schedule with the maximum flow time of at most ℓ. The pseudocode of this algorithm can be found in Algorithm 2 in Section B. The correctness of this algorithm can be easily shown by a simple induction on t.

LEMMA 2.7. *If there exists a schedule for J' with the maximum flow time ℓ, Algorithm 2 yields such a schedule. Otherwise, it declares that the maximum flow time is greater than ℓ.*

We now upper-bound the run time of this algorithm.

LEMMA 2.8. *Algorithm 2 has a run time of at most $O\left(\frac{T}{\epsilon L^*} \cdot (L^*)^2\right) \cdot \left(\frac{3}{\epsilon}\right)^{18L^*} \cdot \left(\frac{6e}{\epsilon}\right)^{\epsilon L^*}$.*

Proof. We start by upper-bounding several quantities. We first bound the size of $\mathcal{Q}^f(t-1)$.

$$|\mathcal{Q}^f(t-1)| \leq |\mathcal{Q}(t-1)| \leq (\ell^2)^{9L^*} \leq \left(\frac{3}{\epsilon}\right)^{18L^*}$$

This follows since by Proposition 2.3, there are at most $9L^*$ pages that can appear in a configuration in $\mathcal{Q}^f(t-1)$ and by Proposition 2.2, each of those pages can be transmitted at most twice during $[t - \ell, t - 1]$.

By Proposition 2.3, the total number of different sets $\mathcal{A}_t$ is upper-bounded by

$$\binom{6L^*}{\epsilon L^*} \leq \left(\frac{6eL^*}{\epsilon L^*}\right)^{\epsilon L^*} = \left(\frac{6e}{\epsilon}\right)^{\epsilon L^*}$$

It is easy to see that the extra overhead for each pair of Q and $\mathcal{A}_t$ is at most $O((L^*)^2)$ (we do not optimize

this, since this is negligible compared to the above two quantities), hence the run time follows. $\square$

Now recall that we have assumed that ϵL^* is an integer. Suppose ϵL^* is not an integer. If $L^* \le 1/\epsilon^2$, by Lemma 2.5, we can find an optimal schedule in time $O(m) \cdot n^{O(1/\epsilon^2)}$. If $L^* \ge 1/\epsilon^2$, then we can find L such that $L^* < L < L^* + 1/\epsilon$ and L is an integral multiple of $1/\epsilon$. Since L approximates L^* within a factor of $1 + \epsilon$, by considering L rather than L^*, we will lose only a multiplicative factor $1 + \epsilon$ in the approximation ratio.

By Lemma 2.6, we derive Theorem 2.1.

2.3 Removing the Dependency of Run Time on T In this section, we will make our algorithm run in polynomial time in m for any fixed $\epsilon > 0$. We consider three cases depending on the value of L^*, and show how the proposed algorithm for each case can be adapted so that the run time does not depend on T. Recall that T is the length of the planning horizon, i.e. an upper-bound on the last time when the optimal schedule transmits. Let $r_{\max} := \max_\rho r_\rho$ denote the latest arrival time of any request. One can without loss of generality set $T := r_{\max} + \min\{m, n\}$. This is because one can satisfy all requests by T by transmitting all outstanding pages after $r_{\max}$ and there is no incentive of transmitting the same page twice when there are no more requests to arrive. Hence if $T = O(m)$, we have nothing to prove. So we will assume that $T > m$. We will say that an interval I is silent if no request arrives during I.

Case (a): $L^* = O(1)$. Recall that Algorithm 1 runs in $O(T) \cdot m^{O(L)}$. Suppose that there is a silent interval $I = [t_1, t_2] \subseteq [1, T]$ of length at least L^*. Suppose that I is maximal. Then since all requests in $\mathcal{R}_{r<t_1}$ must be satisfied by time $t_1 + L^* - 1 (\le t_2)$, one can find an optimal schedule on the requests in $\mathcal{R}_{r<t_1}$ and $\mathcal{R}_{r>t_2}$ separately. Hence one can without loss of generality assume that there is no silent interval of length L^*. This implies that $T = O(L^* \cdot m) = O(m^2)$. Hence we obtain an exact algorithm with run time $m^{O(L^*)}$.

Case (b): $L^* \le \frac{1}{\epsilon^3} \log m$. This case is similar to case (a). In Theorem 2.1, we have shown a $(1 + 6\epsilon)$-approximation with run time $O(\frac{T}{\epsilon L^*}(L^*)^2) \cdot (\frac{3}{\epsilon})^{18L^*} \cdot (\frac{6e}{\epsilon})^{\epsilon L^*}$; here $O(\frac{T}{\epsilon L^*})$ is due to the number of time steps considered in the modified instance J'. Recall that in J', the maximum flow time ℓ was at most $2 + 1/\epsilon$. By a similar argument as for case (a), we can assume that there is no silent interval of length ℓ. Hence it follows $\frac{T}{\epsilon L^*} = O(\ell m)$. Hence we obtain a $(1 + 6\epsilon)$-approximation with run time $O(\frac{m}{\epsilon}) \cdot (\frac{3}{\epsilon})^{18L^*} \cdot (\frac{6e}{\epsilon})^{\epsilon L^*}$.

Case (c): $L^* \ge \frac{1}{\epsilon^3} \log m$. We will show that one can modify $\mathsf{LP}_{\mathsf{MaxFlow}}$ so that the number of time steps (not necessarily continuous) in consideration is at most m.

This will allow us to find an optimal LP solution where $\sum_{p \in \mathcal{P}} x^*_{p,t} \ne 0$ for at most m time steps t. We will show that one can decompose the requests instance into a sub-instance where the number of time steps is as large as the number of requests, and solve each sub-instance separately. We assume without loss of generality that $\mathcal{R}_{r=0} \ne \emptyset$. Suppose that $T > m$, since otherwise there is nothing to prove. Let $t_2 \ge 0$ be the earliest time t such that $|\mathcal{R}_{0 \le r \le t}| \le t + 1$ (or equivalently, $|\mathcal{R}_{0 \le r \le t}| = t + 1$).

CLAIM 2.9. *The optimal schedule (in fact any reasonable schedule that tries to satisfy at least one outstanding request) satisfies all requests $\mathcal{R}_{r \le t_2}$ by time $t_2 + 1$.*

Proof. Without loss of generality assume that the optimal solution transmits a page only if it satisfies at least one outstanding request. If the optimal schedule is never idle by time $t_2 + 1$ (except at time 0), then the claim holds, since $|\mathcal{R}_{0 \le r \le t_2}| = t_2 + 1$. Also if the optimal schedule is idle at time $t_2 + 1$, then the claim again follows, since there are no outstanding requests at time $t_2 + 1$. If not, we show a contradiction. Let t_1 be the latest time no greater than t_2 when the optimal schedule gets idle. Note that $|\mathcal{R}_{t_1 \le r \le t_2}| < t_2 - (t_1 - 1)$. Otherwise, we will have $|\mathcal{R}_{0 \le r \le t_1 - 1}| \le t_1$, which is a contradiction to the definition of t_2. Hence all requests in $\mathcal{R}_{t_1 \le r \le t_2}$ are satisfied during $[t_1 + 1, t_2 + 1]$. Then it implies that the optimal schedule makes $(t_2 - t_1 + 1)$ transmissions to satisfy $t_2 - t_1$ requests. A contradiction.

We have shown that all requests in $\mathcal{R}_{0 \le r \le t_2}$ ($|\mathcal{R}_{0 \le r \le t_2}| = t_2 + 1$) can be satisfied during $[1, t_2 + 1]$. This will become one sub-instance. By repeating this process on the remaining requests $\mathcal{R}_{r > t_2}$, we can identify at most m time steps that we need to consider. Let $\mathcal{T}'$ denote the set of such times. Then we restrict the LP variables only to times in $\mathcal{T}'$. Recall that in Lemma 2.4 we showed that $\Pr[OF(I) \ge 6\epsilon L^*] \le \frac{1}{T^3}$. The proof can be easily modified to show that when $L^* \ge \frac{1}{\epsilon^3} \log m$, $\Pr[OF(I) \ge 6\epsilon L^*] \le \frac{1}{m^3}$. We claim that we only need to focus on the intervals I that start and end at times in $\mathcal{T}'$. To see this, consider any interval $[\tau_1, \tau_2]$. Let $[\tau'_1, \tau'_2]$ be the maximal interval such that $\tau'_1, \tau'_2 \in \mathcal{T}'$, and $[\tau'_1, \tau'_2] \subseteq [\tau_1, \tau_2]$; if no such τ'_1, τ'_2 exist, then $OF([\tau_1, \tau_2]) = 0$, since in σ_{temp}, transmissions can be made only at times in $\mathcal{T}'$. Note that $OF([\tau'_1, \tau'_2]) \ge OF([\tau_1, \tau_2])$. Since there are at most m^2 intervals to consider, by a simple union bound, we obtain a randomized $(1 + 6\epsilon)$-approximation that succeeds with a probability of at least $1 - 1/m$.

From the above three cases, we can show that the run time is $O(\frac{m}{\epsilon}) \cdot (\frac{3}{\epsilon})^{18L^*} \cdot (\frac{6e}{\epsilon})^{\epsilon L^*}$ with $L^* = (1/\epsilon^3) \log m$. By a similar algebra as in the proof of Theorem 2.2, the run time simplifies to $m^{O(1/\epsilon^4)}$, thereby removing the dependency on T.

2.4 Derandomization We derandomize our algorithm in Section 2.1 using the method of pessimistic estimators [32]. This method is by now a standard tool for derandomizing an algorithm that relies on concentration inequalities. It is useful particularly when there is no concrete target value whose expectation is easy to measure, and hence the standard conditional expectation method can not be used. Recall from Section 2.1 that for each interval $I \in \mathcal{I}$, we created a set of 0-1 variables $X_{g,I}$ with $\Pr[X_{g,I} = 1] = v_{g,I} - \lfloor v_{g,I} \rfloor$, and showed that $\Pr\left[\sum_{g \in \mathcal{G}} X_{g,I} \geq \mathbb{E}[\sum_{g \in \mathcal{G}} X_{g,I}] + \Delta\right] \leq \frac{1}{m^3}$, where $\Delta = 6\epsilon L^*$. Let $\mu_{g,I} := v_{g,I} - \lfloor v_{g,I} \rfloor$ and $\mu_I = \sum_{g \in \mathcal{G}} \mu_{g,I}$. (In Section 2.1, to simplify the notation, we considered any fixed interval I and did not include I in the subscript. Also, initially we showed the probability is bounded by $1/T^3$, and in Section 2.3 that we only need to consider at most m^2 intervals and the probability is bounded by $1/m^3$). Recall that for any $g \in \mathcal{G}$, all random variables $\{X_{g,I}\}_{I \in \mathcal{I}}$ are determined by the same random value $\alpha_g \in [0,1]$. For notational convenience, we index groups by integers from 1 to $k := |\mathcal{G}|$ in an arbitrary but fixed way and will refer to groups by their index. To apply the method of pessimistic estimators, we define our "estimator" for each $I \in \mathcal{I}$. We will distinguish intervals in $\mathcal{I}$ into two groups: If $\mu_I \geq \epsilon L^*$ then $I \in \mathcal{I}^{big}$, otherwise $I \in \mathcal{I}^{small}$. Throughout this section, let $\lambda := 1 + 3\epsilon$.

For each $I \in \mathcal{I}^{big}$, define

$$f_I(z_1, z_2, ..., z_k) := \frac{\prod_{i \in [k]} \exp\left((\log \lambda) z_i\right)}{\exp\left(\lambda(\log \lambda)\mu_I\right)}$$

and for each $I \in \mathcal{I}^{small}$, define

$$f_I(z_1, z_2, ..., z_k) := \frac{\prod_{i \in [k]} \exp(z_i)}{\exp(6\epsilon L^*)}$$

We will consider groups in increasing order of their index, and will find a "right" α_i value conditioned on the $\alpha_1, \alpha_2, ..., \alpha_{i-1}$ values that we have found and fixed. To this end, we need to carefully define several quantities. Let $x_{i,I}(\alpha'_i)$ denote $X_{i,I}$'s value when $\alpha_i = \alpha'_i$. When α'_i is clear from the context, we may denote $x_{i,I}(\alpha'_i)$ simply by $x'_{i,I}$. When $\alpha_1 = \alpha'_1, ..., \alpha_i = \alpha'_i$ are fixed, we define the following quantity for each interval $I \in \mathcal{I}$.

$$E_{i,I} := \mathbb{E}[f_I(X_1, X_2, ..., X_k) \mid \alpha_1 = \alpha'_1, ..., \alpha_i = \alpha'_i],$$

where the expectation is defined over $X_{i+1}, X_{i+2}, ..., X_k$ (which are determined by $\alpha_{i+1}, \alpha_{i+2}, ..., \alpha_k$). We note that $E_{i,I}$ can be easily computed in polynomial time since $X_{i,I}, i \in [k]$ are independent.

We will show the following two lemmas.

LEMMA 2.10. *Consider any integer $0 \leq h \leq k$, and any $x_{i,I} \in \{0,1\}$, $i \in [h]$ (more precisely for any $\alpha_{i,I} \in [0,1]$, $i \in [h]$ that determines $x_{i,I}$, $i \in [h]$). Then for any $I \in \mathcal{I}^{big}$,*

$$\Pr\left[\sum_{i \in [k]} X_{i,I} \geq \lambda \mu_I \;\Big|\; X_{i,I} = x_{i,I}, i \in [h]\right] \leq E_{h,I}.$$

Also for any $I \in \mathcal{I}^{small}$,

$$\Pr\left[\sum_{i \in [k]} X_{i,I} \geq 6\epsilon L^* \;\Big|\; X_{i,I} = x_{i,I}, i \in [h]\right] \leq E_{h,I}.$$

LEMMA 2.11. $\sum_{I \in \mathcal{I}} E_{0,I} \leq \frac{1}{m}$.

Lemma 2.10 shows how the probability of the bad event for I (an overflow more than $6\epsilon L^*$) can be bounded by the expectation of our estimator function. Furthermore, this inequality holds for the bad event for I conditioned on any fixed $X_{i,I}$, $i \in [h]$ (or more precisely on fixed α_i, $i \in [h]$). This will allow us to use the conditional expectation method to find a sequence of $\alpha_1 = \alpha'_1$, $\alpha = \alpha'_2, ..., \alpha_k = \alpha'_k$ such that

$$\sum_{I \in \mathcal{I}} E_{k,I} \leq \sum_{I \in \mathcal{I}} E_{k-1,I} \leq ... \leq \sum_{I \in \mathcal{I}} E_{0,I}$$

By Lemma 2.11, we know that $\sum_{I \in \mathcal{I}} E_{k,I} \leq 1/m$. Knowing that $\mu_I \leq 2L^*$, it is easy to see that $\alpha'_1, \alpha'_2, ..., \alpha'_k$ are the desired "good" α values such that for all $I \in \mathcal{I}$,

$$\sum_{i \in [k]} x_{i,I}(\alpha'_i) \leq \mu_I + 6\epsilon L^*.$$

It now remains to prove Lemma 2.10 and 2.11. The proofs are very similar to that of Chernoff inequalities.

Proof of [Lemma 2.10] We first consider any $I \in \mathcal{I}^{big}$. Recall that $\epsilon L^* \leq \mu_I \leq 2L^*$. In the following equations, for notational simplicity, we omit the condition $X_{i,I} = x'_{i,I}, i \in [h]$. Recall that $\lambda = 1 + 3\epsilon$.

$$\Pr\left[\sum_{i \in [k]} X_{i,I} \geq \lambda \mu_I\right]$$

$$= \Pr\left[\exp\left((\log \lambda)\sum_{i \in [k]} X_{i,I}\right) \geq \exp\left((\lambda \log \lambda)\mu_I\right)\right]$$

$$\leq \frac{\mathbb{E}\left[\exp\left((\log \lambda)\sum_{i \in [k]} X_{i,I}\right)\right]}{\exp\left(\lambda(\log \lambda)\mu_I\right)}$$

[By Markov's inequality]

$$= \frac{\prod_{i \in [k]} \mathbb{E}\left[\exp\left((\log \lambda) X_{i,I}\right)\right]}{\exp\left(\lambda(\log \lambda)\mu_I\right)}$$

[Since $X_{i,I}$ are independent]

$$= E_{h,I}$$

We now consider any $I \in \mathcal{I}^{small}$.

$$\Pr\Big[\sum_{i\in[k]} X_{i,I} \geq 6\epsilon L^*\Big]$$

$$= \Pr\Big[\exp\Big(\sum_{i\in[k]} X_{i,I}\Big) \geq \exp\Big(6\epsilon L^*\Big)\Big]$$

$$\leq \frac{\mathbb{E}\Big[\exp\Big(\sum_{i\in[k]} X_{i,I}\Big)\Big]}{\exp(6\epsilon L^*)}$$

$$= \frac{\prod_{i\in[k]} \mathbb{E}\Big[\exp(X_{i,I})\Big]}{\exp(6\epsilon L^*)} = E_{h,I}$$

The last inequality is due to Markov's inequality.
$\square$

Proof of [Lemma 2.10] For each interval $I \in \mathcal{I}$, it suffices to show that $E_{0,I} \leq \frac{1}{m^3}$. We first consider any $I \in \mathcal{I}^{big}$. Recall that $\lambda = 1 + 3\epsilon$ and $L^* \geq (1/\epsilon^3)\log m$.

$$E_{0,I} = \frac{\prod_{i\in[k]}\Big(\exp(\log\lambda)\mu_{i,I} + (1-\mu_{i,I})\Big)}{\lambda^{\lambda\mu_I}}$$

$$= \frac{\prod_{i\in[k]}((\lambda-1)\mu_{i,I} + 1)}{\lambda^{\lambda\mu_I}}$$

$$\leq \frac{\exp(3\epsilon\sum_{i\in[k]}\mu_{i,I})}{(1+3\epsilon)^{(1+3\epsilon)\mu_I}}$$

$$= \Big(\frac{e^{3\epsilon}}{(1+3\epsilon)^{1+3\epsilon}}\Big)^{\mu_I} \leq e^{-3\epsilon^2\mu_I}$$

$$= \frac{1}{m^3}$$

The last inequality holds when $0 \leq \epsilon \leq 1/3$.

Now consider any $I \in \mathcal{I}^{small}$.

$$E_{0,I} = \frac{\prod_{i\in[k]}\Big((e-1)\mu_{i,I} + 1\Big)}{\exp(6\epsilon L^*)}$$

$$\leq \frac{\exp((e-1)\sum_{i\in[k]}\mu_{i,I})}{\exp(6\epsilon L^*)}$$

$$< \exp(-3\epsilon L^*) \leq \frac{1}{m^3}$$

$\square$

2.5 Lower Bound on LP-guided FIFO Since the algorithm FIFO is 2-competitive even in the online setting, one may hope that a natural modification of FIFO may yield a better approximation. In this section, we show that a natural "LP-guided" FIFO does not improve the approximation ratio. The LP-guided FIFO is defined as follows. Let $x^*_{p,t}$ be a fractional solution that satisfies all requests within L^* time steps. At each time t and for any page p, let $y^*_{p,t} := \sum_{t'=t_1+1}^{t} x^*_{p,t'}$ where t_1 is the last time we transmitted page p; if no such time exists, then $t_1 = 0$. Intuitively, $y^*_{p,t}$ suggests the amount or probability that page p needs to be transmitted. Hence using $y_{p,t}$ to determine the priority, we transmit the page p with the largest value of $y^*_{p,t}$ breaking ties favoring the page that has the earliest arriving unsatisfied request.

However, we can find a simple example that shows that this achieves only a 2-approximation. Consider the following instance. We will index pages by integers from 1 to n, and will assume that n is even. At each time $t \in [1, n/2]$, two requests, each for page $2t-1, 2t$, are released. The same sequence of requests is repeated during $[n/2+1, n]$. That is, at each time $t + n/2$ for any $t \in [1, n/2]$, two requests, one for page $2t-1$ and one for page $2t$ are released. This completes the description of the requests instance.

Now consider the following fractional solution $x^*_{p,t}$: For any page p, page p is transmitted three times by half. More precisely, $x_{p,\lceil p/2\rceil+1} = x_{p,\lceil p/2\rceil+n/2+1} = x_{p,\lceil p/2\rceil+n+1} = 1/2$, and for all other times t, $x_{p,t} = 0$. It is easy to check that the maximum flow time is $n/2+1$ in this fractional solution. In contrast, the LP-guided FIFO transmits all pages from 1 to n during $[2, n+1]$, and repeats this during $[n+2, 2n+1]$. In this schedule, the maximum flow time is n.

3 Throughput (Profit) Maximization

In this section, we study the maximum throughput objective in broadcast scheduling. In this setting, each request ρ is associated with its release time r_ρ, deadline d_ρ, weight (profit) w_ρ, and the page p_ρ it asks for. We say request ρ is satisfied within its window if page p is transmitted during $[r_\rho+1, d_\rho]$. If we satisfy request ρ within its window, we obtain the profit w_ρ associated with the request ρ. The goal is to find a schedule that maximizes the total profit.

Our main result is an improved 0.7759-approximation for the maximum throughput objective. More precisely, we give a randomized algorithm with approximation factor $(\alpha_0 - \epsilon)$ for some $\alpha_0 > 1/2 + 3/(4e)$ and any $\epsilon > 0$. Furthermore, the run time is $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$. Note that for any fixed input size, the run time required to achieve an approximation factor arbitrarily close to $(1/2 + 3/(4e))$ increases by a multiplicative factor depending only on ϵ.

The analysis of the 0.7759-approximation is algebraically involved. Hence we first present a 0.754-approximation to illustrate our main ideas behind the improvement. The 0.7759-approximation will be presented in Section 3.3. Also we will first present an algorithm with run time $m^{(1/\epsilon)^{O(1/\epsilon)}}$, and reduce the run time to $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$ in Section 3.4.

Overview of 0.754-**approximation.** Our algorithm improves upon the previous best known approximation by handling requests of large/small windows separately. Here we say that a request ρ has a large window if ρ's window length $|W_\rho| \geq 2H$, otherwise a small window;

later H will be set as $1/\epsilon^3$. Let $\mathcal{R}^{small}$ and $\mathcal{R}^{large}$ denote the set of small-window and large-window requests, respectively.

To introduce our integer/linear programming, we need to define a fair amount of notation. To collect most profits from small-window requests, we create configuration variables $y_{I,Q}$ for each sub-interval I which is defined as follows. Let $I_0, I_1, ..., I_h$ be the minimum number of disjoint intervals that cover the time horizon $[0, T = \max_\rho d_\rho]$ seamlessly; the run time will depend on T, but we will remove this dependency later. All intervals have a length $2H/\epsilon$ except the first interval. We set the length of the first interval as a random number drawn from $[1, 2H/\epsilon]$. In fact, by trying all possible values in $[1, 2H/\epsilon]$, we can assume that we know the right value; what "right" means will become clear soon. Note that the first interval determines all other intervals. We let $\mathcal{I} := \{I_0, I_1, ..., I_h\}$. For each interval I_i, we define configurations. We say that $Q = \{(t, q_t) \mid t \in I_i\}$ is a configuration with respect to I_i, where for all $t \in I_i$, $q_t \in \mathcal{P}$. Of course, the pair (t, q_t) implies that page q_t is transmitted at time t. Let $\mathcal{Q}(I_i)$ denote the collection of all configurations with respect to I_i. For each configuration $Q \in \mathcal{Q}(I_i)$ we create a variable $y_{I_i,Q}$, which is 1 if the schedule follows Q during I_i. Let $w_{I,Q}$ denote the total profit of the small-window requests arriving in I that are satisfied (within their window) by the transmissions made by Q. Also for each request ρ in $\mathcal{R}^{large}$, we create a variable z_ρ that indicates if ρ is satisfied within its window or not. We note that the constant H will depend only on ϵ, therefore the number of variables is at most a polynomial in m and T for any fixed $\epsilon > 0$ (In Section 3.4, we will show how to remove the dependency on T and reduce the run time to $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$).

We are now ready to set up our integer programming. Let $P(Q, I')$ denote the set of pages that are transmitted during I' according to configuration Q.

(IP$_{\mathsf{Throughput}}$)

$$\max \quad \sum_{I \in \mathcal{I}} \sum_{Q \in \mathcal{Q}(I)} w_{I,Q} y_{I,Q} + \sum_{\rho \in \mathcal{R}^{large}} w_\rho z_\rho$$

(3.4)
$$\sum_{Q \in \mathcal{Q}(I)} y_{I,Q} \leq 1 \qquad \forall I \in \mathcal{I}$$

(3.5)
$$\sum_{I \in \mathcal{I}} \sum_{Q \in \mathcal{Q}(I), p_\rho \in P(Q, I \cap W_\rho)} y_{I,Q} \geq z_\rho \qquad \forall \rho \in \mathcal{R}^{large}$$

$$y_{I,Q} \geq 0 \qquad \forall I \in \mathcal{I}, Q \in \mathcal{Q}(I);$$
$$z_\rho \in \{0, 1\} \qquad \forall \rho \in \mathcal{R}^{large}$$

Note that we use different variables $y_{I,Q}$ and z_ρ to count profits from requests in $\mathcal{R}^{small}$ and $\mathcal{R}^{large}$,

respectively. Further, it should be noted that not all the small-window requests contribute to the objective. More precisely, for each interval $I = [t_1, t_2] \in \mathcal{I}$, the small-window requests arriving during $[t_2 - 2H + 1, t_2]$ may not contribute to $\sum_{Q \in \mathcal{Q}(I)} w_{I,Q} y_{I,Q}$. This can happen if the optimal solution satisfies some of those requests during the next interval I' starting at $t_2 + 1$. Because of this, we may lose some profits for those requests. However, for the "right" choice of the length of the first interval in $\mathcal{I}$, we will lose only ϵ fraction of profits from small-window requests, since small-window requests are discarded from only ϵ fraction of time steps.

Constraints (3.4) restrict that only one configuration can be selected for each interval $I \in \mathcal{I}$. Constraints (3.5) follow from the definition of z_ρ. A large-window request ρ is satisfied if p_ρ is transmitted during $I \cap W_\rho$ for some interval $I \in \mathcal{I}$. We can relax this integer program into a linear program $\mathsf{LP_{Throughput}}$ by replacing $z_\rho \in \{0, 1\}$ with $0 \leq z_\rho \leq 1$. We let $x_{p,t} = \sum_{Q:(t,p) \in Q, Q \in \mathcal{Q}(I)} y_{I,Q}$ for all $I \in \mathcal{I}, t \in I, p \in \mathcal{P}$, which will relate configuration-based variables y to time-indexed variables x.

We can obviously solve $\mathsf{LP_{Throughput}}$ in polynomial time for any fixed precision factor constant ϵ. Since for each $I \in \mathcal{I}$, there can be at most $m^{2H/\epsilon} = m^{O(1/\epsilon^4)}$ variables, the run time will be also $m^{O(1/\epsilon^4)}$. As mentioned before, we will show how to reduce the run time down to $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$ in Section 3.4. We will let OPT denote the optimal fractional solution of the above LP. We will simply try all possible sizes of the first interval in $\mathcal{I}$, and will take the best size that maximizes the objective in $\mathsf{LP_{Throughput}}$. It is easy to see that OPT is at least $(1 - \epsilon)$ times as large as the integral optimum. Let OPT^{small} and OPT^{large} denote the amount that small-window and large-window requests contribute to OPT, respectively (the first and second sum in the objective, respectively).

We give a randomized rounding that achieves an expected total profit of at least $(1-\epsilon)^2 \cdot (1-2/3e)\mathsf{OPT}$ for any $\epsilon > 0$. We develop two different rounding schemes and combine them. We first give a high-level overview of the algorithm and how we get an improved approximation factor. When OPT^{small} is relatively large, we use an independent rounding that selects one configuration $Q \in \mathcal{Q}(I)$ with probability $y_{I,Q}$ for each $I \in \mathcal{I}$. Intuitively, since configurations consider enough details for satisfying small-window requests, we will be able to collect most profits from small-window requests. At the same time, due to the nature of independent rounding, we will be able to collect $(1-1/e)\mathsf{OPT}^{large}$ from large-window requests. In Appendix 3.1, we prove that we can achieve a total expected profit of at least

(3.6) $$\mathsf{OPT}^{small} + (1 - 1/e)\mathsf{OPT}^{large}$$

On the other hand, when OPT^{large} is relatively large,

we use a different rounding scheme by modifying the α-point rounding. Unlike the independent rounding, the α-point rounding does not immediately yield a feasible schedule since there could be some time slots where more than one page is transmitted. Hence we need a certain contention resolution scheme. Here the difficulty is that if we move some transmissions to near time slots, they may become completely useless. This is because in the throughput objective, we can obtain a profit w_ρ only when we can satisfy ρ within its window. This could be the case particularly for small-window requests. Hence in this setting, we do not collect as much profits from small-window requests as when using the independent rounding scheme. Nevertheless, we will keep only one page using the fair contention resolution scheme in [22], and will obtain a $(1 - 1/e)$-approximation, which is the same approximation guarantee that the independent rounding gives. For large-window requests, we will be able to collect more profits than $(1-1/e)\mathsf{OPT}^{large}$ since large-window requests are less fragile to moving transmissions. In Section 3.2, we show that we can collect a total expected profit of at least

(3.7)
$$(1-\epsilon)^2((1-1/e)\mathsf{OPT}^{small} + (1-1/2e)\mathsf{OPT}^{large})$$

Assuming that ϵ is arbitrarily small, the minimum of the above two quantities (3.6) and (3.7) is achieved when $\mathsf{OPT}^{large} = 2\mathsf{OPT}^{small}$. Hence by selecting the better of these two solutions, we achieve a profit of at least $(1-\epsilon)^2(1-2/3e)\,\mathsf{OPT}$.

Since the LP relaxation loses at most ϵ fraction of total profit from small-window requests compared to the optimal solution, we derive Theorem 1.2.

3.1   When Small-window Requests Give Large Profits As mentioned previously, for each $I \in \mathcal{I}$, we pick one configuration $Q \in \mathcal{Q}(I)$ with probability $y_{Q,I}$. By (3.4) and from the objective, we know that the expected profit we obtain from small-window requests is at least OPT^{small}. Now consider any large-window request ρ. Let $I'_1, ..., I'_k \in \mathcal{I}$ denote the time intervals that ρ's window spans over, i.e. $r_\rho + 1 \in I'_1$ and $d_\rho \in I'_k$. Let $\gamma_i := \sum_{Q:Q\in\mathcal{Q}(I'_i):p_\rho\in P(Q,I'_i\cap W_\rho)} y_{Q,I'_i}$ denote the fraction of configurations with respect to I'_i that satisfy request ρ. Note that we have $z_\rho = \min\{\sum_{i\in[k]}\gamma_i, 1\}$ due to constraints (3.4) and (3.5).

Hence the probability that request ρ is satisfied is at least

$$1 - \prod_{i=1}^{k}(1-\gamma_i) \geq 1 - \prod_{i=1}^{k}\exp(-\gamma_i)$$
$$= 1 - \exp(-\sum_{i=1}^{k}\gamma_i) \geq 1 - \exp(-z_\rho)$$
$$\geq (1-1/e)z_\rho$$

This gives us $(1 - 1/e)\mathsf{OPT}^{large}$ profit in expectation from large-window requests. This proves the profit claimed in (3.6).

3.2   When Large-window Requests Give Large Profits This section is devoted to describing and analyzing an algorithm that achieves a profit as large as (3.7). This will be useful when large-window requests give a relatively large total profit.

3.2.1   Algorithm Our algorithm uses only $x_{p,t}$ and consists of several steps.

1. α-point Rounding: The first step using the standard α-point rounding we obtain an infeasible tentative schedule. Formally, for each page p, choose α_p uniformly at random from $[0, 1]$. Then transmit page p at all times t such that $\sum_{t'=1}^{t-1} x_{p,t'} < \alpha_p + k \leq \sum_{t'=1}^{t} x_{p,t'}$ for some integer $k \geq 0$. Let A_t denote the set of all pages transmitted at time t, and let σ_{temp} denote the resulting infeasible schedule; $|A_t| > 1$ could occur for some time t.

2. Fair Contention Resolution: In this step, the goal is to keep at most one page p_t from A_t at each time t; if $A_t = \emptyset$, p_t may not exist. We will say that p_t is the first-round page transmitted at time t. We use the fair contention resolution from [22], and choose a page p from A_t with probability

$$\frac{1}{\sum_{q\in\mathcal{P}} x_{q,t}}\Big(\sum_{q\in A_t\setminus\{p\}} \frac{x_{q,t}}{|A_t|-1} + \sum_{q\in\mathcal{P}\setminus A_t}\frac{x_{q,t}}{|A_t|}\Big)$$

By Lemma 1.4 in [22], it follows that

$$\Pr[p_t = p] \geq \frac{1 - \prod_{q\in\mathcal{P}}(1-x_{q,t})}{\sum_{q\in\mathcal{P}} x_{q,t}}\cdot x_{p,t} \geq (1-1/e)x_{p,t},$$

since $\sum_{q\in\mathcal{P}} x_{q,t} \leq 1$.

To get more profits from large-window requests, we will try to reschedule pages in $A_t \setminus \{p_t\}$ to near time slots so that those pages can still satisfy some large-window requests that they could before rescheduling; we abuse the notation $\{p_t\}$ by letting it denote $\emptyset$ if p_t does not exist. To make sure those pages can find empty time slots nearby (with a probability close to 1), we empty a small fraction of time slots. These freed-up time slots, together with time slots t such that $|A_t| = 0$, will be used as new places to accommodate overflowed transmissions.

3. Freeing-up Times: Pick a random value h uniformly at random from $[1, \epsilon H]$. Let $\mathcal{B} := \{B_i, 0 \leq i \leq \lceil T/H\rceil+1\}$ denote a set of disjoint intervals where $B_0 := [0, h)$, and $B_i := [h+(i-1)\epsilon H, h+i\epsilon H)$ for all $i \geq 1$. Note that all intervals in $\mathcal{B}$ have the same length except the first interval B_0. To distinguish the intervals in $\mathcal{B}$ from the intervals in $\mathcal{I}$ used in defining $\mathsf{LP}_{\mathsf{Throughput}}$, we will call the intervals

in $\mathcal{B}$ blocks. Note that the blocks in $\mathcal{B}$ cover the interval $[0, T]$ seamlessly. We now empty the last $\epsilon^2 H$ time slots from each block in $\mathcal{B}$ by discarding all pages in A_t at those times. We will say that those time slots are "freed-up."

4. Relocating Overflowed Transmissions (pages): The goal of this step is to with a good probability relocate all pages in $A_t \setminus \{p_t\}$ at non-freed-up times t to nearby empty time slots. By empty time slots, we mean the "freed-up" times plus the empty times t from the beginning, i.e. t such that $|A_t| = 0$ in σ_{temp}. We first decide in which direction to move overflowed transmissions. The direction will be decided either as right or left, each with a probability $1/2$. Let $\mathtt{Right}$ and $\mathtt{Left}$ denote the former and the latter event, respectively. All overflowed transmissions follow the same direction. Formally, for each block $B = [t_1, t_2 + \epsilon^2 H) \in \mathcal{B}$ (if $t_2 < 0$, do nothing), we do the following: For each time $t \in [t_1, t_2)$ and $p \in A_t$,

- if $p = p_t$, the page p stays at time t.

- if $p \neq p_t$ and the event $\mathtt{Left}$ occurs, the page p can move to an empty time in $[t_1 - \epsilon^2 H, t)$.

- if $p \neq p_t$ and the event $\mathtt{Right}$ occurs, the page p can move to an empty time in $[t + 1, t_2 + \epsilon^2 H)$.

Since this is the final schedule, at most one page can be transmitted at a time. Note that the perfect relocation for a block B corresponds to a matching where all the transmissions at non-freed-up times in B are covered by time steps in the interval $[t_1 - \epsilon^2 H, t_2)$ or $[t_1, t_2 + \epsilon^2 H)$ (which is determined by the event $\mathtt{Left}$ or $\mathtt{Right}$). We try to find such a matching. If unsuccessful, we keep only the first-round pages, i.e. $p_t, t \in [t_1, t_2)$ and throw out all other overflowed pages, i.e. $A_t \setminus \{p_t\}, t \in [t_1, t_2)$. This will suffice to give our claimed approximation guarantee.

3.2.2 Analysis We say that page p originates from time t if it is transmitted at time t (in σ_{temp}) by the α-point rounding. When a page originates from t, it may stay at the time t ($p_t = p$), or may be relocated to near time slots. Also it may be discarded when it fails to find a new place. Let $\mathtt{Fst}(\rho, t)$ denote the event that ρ is satisfied by the page p_ρ that originates from time t, and stays at the time t. Let $\mathtt{Snd}(\rho, t)$ denote the event that ρ is satisfied by the page p_ρ that originates from time t but is successfully relocated to other time slots.

Throughout this section, we will consider all large-window requests $\mathcal{R}^{large}$. Additionally we will also consider all small-window requests whose window is completely contained in an interval in $\mathcal{I}$. For notational simplicity, by $\mathcal{R}^{small}$, we will refer to only such small-window requests. Recall that we lose only ϵ fraction of small-window requests. For a request $\rho \in \mathcal{R}^{small}$, z_ρ

is defined as $\min\{\sum_{t \in W_\rho} x_{p_\rho, t}, 1\}$. We note that for any request ρ, $\sum_{t \in W_\rho} x_{p_\rho, t} \geq \sum_{I \in \mathcal{I}} z_{\rho, I}$; recall that $z_{\rho, I} := \sum_{Q \in \mathcal{Q}(I), p_\rho \in P(Q, I)} y_{I, Q}$. Both sides may not be equal when a configuration $Q \in \mathcal{Q}(I)$ transmits page p_ρ several times in $W_\rho \cap I$. Then the configuration view thinks of it as satisfying ρ once while the time indexed view thinks of it as satisfying ρ multiples times. There is no difference between these two views in the integer programming, but they can differ in linear programming relaxation. Our achieved profit for small-window requests will be compared against the time-indexed LP profit, namely, $\sum_\rho w_\rho \cdot \min\{\sum_{t \in W_\rho} x_{p_\rho, t}, 1\}$. This is justified since it is always no smaller than $\mathsf{OPT}^{small} = \sum_{I \in \mathcal{I}} \sum_{Q \in \mathcal{Q}(I)} w_{I, Q} y_{I, Q}$.

We will show the following two lemmas. As discussed in the algorithm's description, the first lemma easily follows from the property of the Fair Contention Resolution scheme in step (2).

LEMMA 3.1. *For all requests ρ and any time t, $\Pr[\mathtt{Fst}(\rho, t) \mid p_\rho \in A_t] \geq (1 - \epsilon)(1 - 1/e)$.*

LEMMA 3.2. *Suppose $H \geq 1/\epsilon^3$ and $0 < \epsilon \leq 1/100$. Then for all large-window requests $\rho \in \mathcal{R}^{large}$ and any time t, $\Pr[\mathtt{Snd}(\rho, t) \mid \neg \mathtt{Fst}(\rho, t) \text{ and } p_\rho \in A_t] \geq (1 - \epsilon)/2$.*

Throughout the analysis, we make a simplifying assumption that $\sum_{t \in W_\rho} x_{p_\rho, t} \leq 1$. Otherwise, it will only help our analysis. Since all the events that $p_\rho \in A_t, t \in W_\rho$ are disjoint, these two lemmas will imply that a large-window request ρ is satisfied within its window with a probability of at least

$$\sum_{t \in W_\rho} \Pr[\mathtt{Fst}(\rho, t) \text{ or } \mathtt{Snd}(\rho, t) \mid p_\rho \in A_t] \cdot \Pr[p_\rho \in A_t]$$

$$= \sum_{t \in W_\rho} \Big(\Pr[\mathtt{Fst}(\rho, t) \mid p_\rho \in A_t] +$$

$$\Pr[\mathtt{Snd}(\rho, t) \text{ and } \neg \mathtt{Fst}(\rho, t) \mid p_\rho \in A_t] \Big) \cdot \Pr[p_\rho \in A_t]$$

$$= \sum_{t \in W_\rho} \Big(\Pr[\mathtt{Fst}(\rho, t) \mid p_\rho \in A_t] + \Pr[\mathtt{Snd}(\rho, t) \mid$$

$$\neg \mathtt{Fst}(\rho, t) \text{ and } p_\rho \in A_t](1 - \Pr[\mathtt{Fst}(\rho, t) \mid p_\rho \in A_t]) \Big)$$

$$\cdot \Pr[p_\rho \in A_t]$$

$$\geq \sum_{t \in W_\rho} (1 - \epsilon)^2 (1 - 1/2e) x_{p_\rho, t}$$

$$= (1 - \epsilon)^2 (1 - 1/2e) z_\rho$$

(3.8)

Also we can easily show from Lemma 3.1 that any request ρ is satisfied with a probability of at least $\sum_{t \in W_\rho} \Pr[\mathtt{Fst}(\rho, t)] \geq (1 - \epsilon)(1 - 1/e) z_\rho$. Hence we will be able to derive the profit claimed in (3.7).

It now remains to show Lemma 3.2.

Proof of [Lemma 3.2] Consider any large-window request ρ and time t'. Throughout, we assume that the time t' is not freed-up, which is the case with probability $(1 - \epsilon)$. Also assume that $p_\rho \in A_{t'}$ but $p_{t'} \neq p_\rho$. The proof will be conditioned on these events.

We first consider the event $\texttt{Right}$. Let $B = [t_1, t_2 + \epsilon^2 H) \in \mathcal{B}$ be the block that includes time t'. We begin with showing the following claim.

CLAIM 3.3. *If for all times $t \in [t_1, t_2)$, $\sum_{\tau \in [t, t_2)} |A_\tau| \leq t_2 - t + \epsilon^2 H$, then we can relocate each overflowed page $p \in A_t \setminus \{p_t\}$, $t \in [t_1, t_2)$, to the right, to one of the empty time slots in B (including p_ρ originating from t').*

Proof. We consider overflowed pages at time t from t_2 to t_1, and move each of such pages to the right most empty time slot in B. By an induction on time t, we can show that all overflowed pages in $A_\tau \setminus \{p_\tau\}$, $t \leq \tau \leq t_2 - 1$ are relocated to an empty time slot in $(t, t_2 + \epsilon^2 H)$. This is obviously true when $t = t_2$ (t_2 was freed-up, so no overflow at time t_2). To complete the induction, it suffices to show that there are enough time slots in $[t, t_2 + \epsilon^2 H)$ to accommodate all transmissions made in σ_{temp} during $[t, t_2)$, and this is exactly what the condition of the claim implies. □

Let $\texttt{Bad}_t$ denote the event that $\sum_{\tau \in [t, t_2)} |A_\tau| > t_2 + \epsilon^2 H - t$. We will show that $\Pr[\texttt{Bad}_t] \leq \epsilon/H$. Then by taking a union bound over all $t \in [t_1, t_2)$, we can show that the desired relocation can be done with a probability of at least $1 - \epsilon$. Bounding the probability of this bad event is very similar to the proof of Lemma 2.4. Let X_p denote a 0-1 random variable that has value one with probability $\mu_p := \sum_{t \in [t, t_2)} x_{p,t} - \lfloor \sum_{t \in [t, t_2)} x_{p,t} \rfloor$. Let $\mu := \sum_{p \in \mathcal{P}} \mu_p$. Note that

$$\Pr[\texttt{Bad}_t] \leq \Pr[\sum_{p \in \mathcal{P}} X_p \geq \mu + \epsilon^2 H]$$

Also note that $\texttt{Var}[\sum_{p \in \mathcal{P}} X_p] = \sum_{p \in \mathcal{P}} \texttt{Var}[X_p] \leq \sum_{p \in \mathcal{P}} (\mu_p - \mu_p^2) \leq \sum_{p \in \mathcal{P}} \sum_{t \in [t, t_2)} x_{p,t} \leq H$. Hence by applying Theorem A.1 with $V \leq H$, $b \leq 1$, and $\Delta = \epsilon^2 H$, we have

$$\Pr[\texttt{Bad}_t] \leq \exp(-\frac{\Delta^2}{2V + 2b\Delta/3}) \leq \exp(-\epsilon^2 H/3)$$
$$\leq \epsilon/H,$$

when $H \geq 1/\epsilon^3$ and $\epsilon \leq 1/100$. Hence we know that in the event $\texttt{Right}$, all overflowed pages in B can be safely relocated with a probability of at least $(1 - \epsilon)$. One can easily show that this is the case also for the event $\texttt{Left}$. The only exception is when B is the first block B_0 in $\mathcal{B}$, since there are no empty slots before time 0.

Suppose B is not the first block. Since ρ is a large-window request (of length at least $2H$), its window W_ρ must cover at least one of $t_1 - \epsilon^2 H$ or $t_2 + \epsilon^2 H$. Hence the request ρ is satisfied by the page p_ρ originating from time t' for at least one of the two events $\texttt{Left}$ and $\texttt{Right}$, which move the page to $[t_1 - \epsilon^2 H, t')$ and $(t', t_2 + \epsilon^2 H)$, respectively. This implies that the event $\texttt{Snd}(\rho, t')$ occurs with a probability of at least $(1 - \epsilon)/2$. Now suppose B is the first block. In this case, any large-window request ρ that starts during the first block covers the entire second block. Hence in the event $\texttt{Right}$, if the relocation is successful, the request will be satisfied. □

3.3 Further Improvement: 0.7759-approximation

In this section we further improve our approximation guarantee. The algorithm remains almost the same (except small changes in preprocessing), but the improvement comes from a more refined analysis. We will give a 0.7759-approximation, more precisely a randomized $(1/2 + 3/(4e) - \epsilon)$-approximation for any $\epsilon > 0$. The main idea is to collect more profits from large-window requests. In the previous analysis we counted profits from small-window/large-window requests separately. Recall that by using the α-point rounding together with the contention resolution scheme, we were able to satisfy a large-window request ρ in the second round with about a half probability if it is not satisfied in the first round (see Lemma 3.2). The reason for the half probability loss can be summarized as follows: When the α-point rounding tries to transmit page p_ρ near to ρ's boundary, the transmission may be relocated to a time step which is out of ρ's window. Since we moved the transmission either to the left or right, each with a half probability, the half loss could occur. However, if ρ is satisfied in the "middle" of its window, we will be able to avoid such a loss. Hence our analysis will consider two cases depending on whether ρ is satisfied near to its boundary or not.

To this end, we preprocess large-window requests as well. Recall that the time horizon was divided into intervals $\mathcal{I} = \{I_0, I_1, ..., I_h\}$ where all intervals have length $2H/\epsilon$ except the first. For each large-window request ρ, we divide its window into three sub-windows, W_ρ^l, W_ρ^m and W_ρ^r: Consider all intervals in $\mathcal{I}$ that intersect W_ρ, and W_ρ^l and W_ρ^r are the intersections with the first and last of those intervals, respectively. The middle window $W_\rho^m := W_\rho \setminus (W_\rho^l \cup W_\rho^r)$ is defined as the remaining interval of W_ρ other than W_ρ^l and W_ρ^r. (If W_ρ intersects only one interval in I_ρ, then $W_\rho^l := w_\rho$, and there are no W_ρ^m, W_ρ^r. If W_ρ intersects exactly two intervals, there is no middle window for ρ). Recall that we discarded all small-window requests that are not fully contained in an interval in $\mathcal{I}$. We also discard some large-window requests. We discard a large-window request ρ if W_ρ^l or W_ρ^r has

a length less than $2\epsilon H$; every large-window request has a window of length at least $2H$. Note that we lose only $4\epsilon H/(2H) \leq 2\epsilon$. For simplicity we assume that $\epsilon \leq 1/2$, and hence we lose only ϵ fraction of total profit from large-window requests in expectation. We let $W_\rho^b := W_\rho^l \cup W_\rho^r$ to denote possibly two "boundary" windows together.

For any $I \in \mathcal{I}$, configuration $Q \in \mathcal{Q}(I)$, and for any interval I', we let $P(Q, I')$ denote the set of pages transmitted during I' in the configuration Q. Let $z_{\rho,I} := \sum_{Q \in \mathcal{Q}(I), p_\rho \in P(Q, W_\rho \cap I)} y_{I,Q}$ denote the total fraction of configurations in the LP that satisfies ρ in I. Let $z_\rho^m := \min\{\sum_{I \in \mathcal{I}, I \in W_\rho^m} z_{\rho,I}, 1\}$, and let $z_\rho^b := \min\{\sum_{I \in \mathcal{I}, I \in W_\rho} z_{\rho,I}, 1\} - z_\rho^m$. Notice that $z_\rho^m + z_\rho^b = z_\rho$. We split OPT^{large} into two quantities, $\mathsf{OPT}_{boundary}^{large} := \sum_\rho w_\rho z_\rho^b$ and $\mathsf{OPT}_{middle}^{large} := \sum_\rho w_\rho z_\rho^m$. Note that $\mathsf{OPT}^{large} = \mathsf{OPT}_{boundary}^{large} + \mathsf{OPT}_{middle}^{large}$. By the independent rounding we will show that we can get a total expected profit of at least

$$(3.9) \quad \mathsf{OPT}^{small} + \frac{2}{e}\mathsf{OPT}_{boundary}^{large} + \frac{3}{2e}\mathsf{OPT}_{middle}^{large}$$

To have a feel how we get these constants $2/e$ and $3/(2e)$, suppose that a large-window request is satisfied only in its boundary windows W_ρ^l or W_ρ^r, by at most one unit in total. Let b_1 and b_2 denote the amount of configurations that transmit page p_ρ in W_ρ^l and W_ρ^r, respectively. Then we know that $b_1 + b_2 = z_\rho \leq 1$. The probability that request ρ is satisfied is then $1 - (1 - b_1)(1 - b_2) = b_1 + b_2 - b_1 b_2 \geq b_1 + b_2 - (b_1 + b_2)^2/4 \geq (3/4)(b_1 + b_2) = (3/4)z_\rho$. In another extreme case when ρ is satisfied only during ρ's middle window, we know that ρ is satisfied with a probability of at least $(1 - 1/e)z_\rho$. Ideally, it would be great if we could obtain $(3/4)\mathsf{OPT}_{boundary}^{large} + (1 - 1/e)\mathsf{OPT}_{middle}^{large}$. However, it is not the case when ρ is satisfied both in W_ρ^b and W_ρ^m. Nevertheless, by careful analysis we will be able to show (3.9) in Section 3.3.1.

Also using the variant of α-point rounding with the contention resolution scheme we will show that we can achieve a total profit of at least
$$(3.10)$$
$$(1 - \epsilon)^2 \Big((1 - \frac{1}{e})\mathsf{OPT}^{small} + (1 - \frac{1}{2e})\mathsf{OPT}_{boundary}^{large} + \mathsf{OPT}_{middle}^{large}\Big)$$

As discussed above, if the α-point rounding tries to satisfy a large-window request ρ in the middle of ρ's window, then we can almost always satisfy ρ within its window, either in the first round or second round.

By selecting the better of these two outcomes, we can get a total expected profit of at least the average of the two,

hence we get an approximation guarantee of

$$(1 - \epsilon)^2 \cdot \Big((1 - \frac{1}{2e})\mathsf{OPT}^{small} + (\frac{1}{2} + \frac{3}{4e})\mathsf{OPT}^{large}\Big)$$
$$\geq \quad (1 - \epsilon)^2 \cdot (\frac{1}{2} + \frac{3}{4e})\mathsf{OPT} \simeq 0.7759(1 - \epsilon)^2\mathsf{OPT}$$

3.3.1 Approximation Guarantee of Independent Rounding: Proof of (3.9)

In this section, we show that the independent rounding of picking one configuration $Q \in \mathcal{Q}(I)$ with probability $y_{Q,I}$ gives an expected total profit claimed in (3.9). As before, we can show that we get an expected profit of OPT^{small} from small-window requests. Hence we focus on proving that for each large-window request ρ (which was not discarded in the beginning), we get an expected profit of at least

$$(3.11) \quad\quad (2/e)z_\rho^b + (3/(2e))z_\rho^m$$

By summing this lower bound over all large-window requests ρ, we can prove the profit claimed in (3.9) due to linearity in expectation.

Recall that $P(Q, I')$ denotes the pages that are transmitted during interval I' by configuration Q. Also recall $z_{\rho,I} := \sum_{Q \in \mathcal{Q}(I), p_\rho \in P(Q, W_\rho \cap I)} y_{I,Q}$ denotes the total fraction of configurations in the LP that satisfies ρ in I. For each large window request ρ we define the following three quantities (for notational convenience we omit ρ from the notation):

$$\eta_1 := z_{\rho,I} \quad\quad \text{for the unique } I \in \mathcal{I} \text{ s.t. } W_\rho^l \subseteq I$$
$$\eta_2 := \sum_{I \in \mathcal{I}: I \subseteq W_\rho^m} z_{\rho,I}$$
$$\eta_3 := z_{\rho,I} \quad\quad \text{for the unique } I \in \mathcal{I} \text{ s.t. } W_\rho^r \subseteq I$$

Intuitively, these quantities η_1, η_2 and η_3 represent how much the LP satisfies a large window request ρ in its left, middle, right windows, respectively. Observe that $\eta_1, \eta_3 \leq 1$, but not necessarily $\eta_2 \leq 1$. Note that $z_\rho^b = \min\{\eta_1 + \eta_2 + \eta_3, 1\} - z_\rho^m$ and $z_\rho^m = \min\{\eta_2, 1\}$. The probability that ρ is satisfied in its window W_ρ is at least

$$\begin{aligned} &(1 - (1 - \eta_1)(1 - \eta_3)) \\ &+ (1 - \eta_1)(1 - \eta_3)(1 - \Pi_{I \in \mathcal{I}: I \subseteq W_\rho^m}(1 - z_{\rho,I})) \\ \geq\ &(1 - (1 - \eta_1)(1 - \eta_3)) \\ &+ (1 - \eta_1)(1 - \eta_3)(1 - \exp(-\eta_2)) \\ \geq\ &(1 - (1 - \eta_1)(1 - \eta_3)) \\ &+ (1 - \eta_1)(1 - \eta_3)(1 - \exp(-z_\rho^m)) \\ \geq\ &(\eta_1 + \eta_3 - \eta_1\eta_3)\exp(-z_\rho^m) + 1 - \exp(-z_\rho^m) \end{aligned}$$

We will show the following lemma which will show (3.11), thereby completing the proof of (3.9). The proof is fairly algebraic.

LEMMA 3.4. *It holds that* $(\eta_1 + \eta_3 - \eta_1\eta_3)\exp(-z_\rho^m) + 1 - \exp(-z_\rho^m) \geq \frac{2}{e}z_\rho^b + \frac{3}{2e}z_\rho^m.$

Proof. For any fixed values of $\eta_1 + \eta_3$ and η_2 (which fix z_ρ^b and z_ρ^m), observe the left-hand-side is minimized when $\eta_1 = \eta_3$. Hence it suffices to show that

$$(\eta_1 + \eta_3 - (\eta_1 + \eta_3)^2/4)\exp(-z_\rho^m) + 1 - \exp(-z_\rho^m)$$
$$\geq \frac{2}{e}z_\rho^b + \frac{3}{2e}z_\rho^m$$

Also since $(\eta_1 + \eta_3 - (\eta_1 + \eta_3)^2/4)$ increases in $\eta_1 + \eta_3$ (when $0 \leq \eta_1 + \eta_3 \leq 2$), and $z^b \leq \eta_1 + \eta_3$, it suffices to show that

$$(3.12) \quad (z_\rho^b - (z_\rho^b)^2/4)\exp(-z_\rho^m) + 1 - \exp(-z_\rho^m)$$
$$\geq \frac{2}{e}z_\rho^b + \frac{3}{2e}z_\rho^m$$

For notional convenience, let $x := z_\rho^b$ and $y := z^m b_\rho$ (These variables have nothing to do with the variables in the LP. This override will be in effect only in the proof of this lemma). Note that $x + y \leq 1$ and $x, y \geq 0$. By rearranging terms in (3.12) it remains to show

$$g(x, y) := (1 - \frac{x}{2})^2 e^{-y} + \frac{2}{e}x + \frac{3}{2e}y \leq 1$$

Observe that for any fixed x, the function $g(x, y)$ is maximized when

- Case (i) $y = 0$; or
- Case (ii) $\frac{\partial g}{\partial y} = 0$; or
- Case (iii) $y = 1 - x$.

We continue our analysis by considering these cases as follows.

Case (i): First consider the case when $y = 0$. Then it is easy to see that $g(x, 0) = (1 - x/2)^2 + (2/e)x$ is maximized when $x = 0$ for any $0 \leq x \leq 1$, and $g(0, 0) = 1$, hence $g(x, 0) \leq 1$.

Case (ii): Now consider the case when $\frac{\partial g}{\partial y} = 0$, which yields $(1 - x/2)^2 e^{-y} = 3/(2e)$. Using this we have $g(x, y) = \frac{3}{2e}(1 + y) + \frac{2}{e}x$. Also we observe that $y + 1 \leq e^y \leq \frac{2e}{3}(1 - \frac{x}{2})^2$. Hence we derive that $g(x, y) = \frac{3}{2e}(1 + y) + \frac{2}{e}x \leq (1 - \frac{x}{2})^2 + \frac{2}{e}x$. As in the previous case, this is at most 1 for any $x \in [0, 1]$.

Case (iii): It now remains to consider the final case when $y = 1 - x$. By plugging this in $g(x, y)$ we derive that

$$g(x) := g(x, 1 - x) = (\frac{x}{2} - 1)^2 \exp(x - 1) + \frac{1}{2e}x + \frac{3}{2e}$$

By simple calculations we have

$$g'(x) := \frac{d}{dx}g(x) = \frac{x}{2}(\frac{x}{2} - 1)e^{x-1} + \frac{1}{2e}$$
$$g''(x) := \frac{d^2}{d^2x}g(x) = (\frac{x^2}{4} - \frac{1}{2})e^{x-1}$$

Since $g''(x) < 0$ for any $x \in [0, 1]$, $g'(0) > 0$ and $g'(1) < 0$, there exists a unique $x_0 \leq [0, 1]$ such that function $g(x)$ increases during $[0, x_0)$ and decreases during $(x_0, 1]$. From the facts that $g(0.75) < 0.993994$, $g(0.755) > 0.993996$, and $g(0.76) \leq 0.993994$, we derive that $0.75 \leq x_0 \leq 0.76$.

Hence we conclude that

$$\max_{0 \leq x \leq 1} g(x) = g(x_0)$$
$$< (\frac{0.75}{2} - 1)^2 \cdot \exp(0.76 - 1) + \frac{1}{2e} \cdot 0.76 + \frac{3}{2e}$$
$$< 0.99889 < 1$$

This completes the proof of the lemma.

3.3.2 Approximation Guarantee of the algorithm in Section 3.2.1: Proof of (3.10)

As before we assume that $\sum_{t \in W_\rho} x_{p_\rho, t} \leq 1$, since otherwise it can only improve our approximation guarantee. Now we take a closer look at the proof of Lemma 3.2, and observe the following:

LEMMA 3.5. *Suppose $H \geq 1/\epsilon^3$ and $0 < \epsilon \leq 1/100$. Then for all large-window requests $\rho \in \mathcal{R}^{large}$ and any time t, we have*

$$\Pr[\mathit{Snd}(\rho, t) \mid \neg \mathit{Fst}(\rho, t) \text{ and } p_\rho \in A_t \text{ and } t \in W_\rho^m]$$
$$\geq (1 - \epsilon)/2$$

and

$$\Pr[\mathit{Snd}(\rho, t) \mid \neg \mathit{Fst}(\rho, t) \text{ and } p_\rho \in A_t \text{ and } t \in W_\rho^b]$$
$$\geq (1 - \epsilon)$$

Proof. The proof is very similar to that of Lemma 3.2, hence we give a sketch of the proof highlighting differences. The proof of the first probability remains the same. The reason why we have a better probability for the second case is the following. Recall that in the proof of Lemma 3.2, we moved a overflowed transmission to the right or left by at most ϵH time steps. Any large-window request ρ we did not discard in the beginning (when solving the LP) intersects each of the two boundary windows W_ρ^l and W_ρ^r by at lest $2\epsilon H$ time steps. Also recall that we have shown that when we try to move the "overflowed" transmissions in an interval $I \in \mathcal{I}$ to the right or left, we can find enough empty time slots for those transmissions with a probability of at least $(1 - \epsilon)$. Hence regardless of the direction in which the transmission (p_ρ, t) is moved, it finds an empty time slot with a probability of at least $(1 - \epsilon)$ and still satisfies ρ in its window conditioned on the transmission not being chosen in the first round. □

We are now ready to complete the proof of (3.10). Since all the events that $p_\rho \in A_t, t \in W_\rho$ are disjoint, the two bounds in the lemma will imply that a large-window

request ρ is satisfied within its window with a probability of at least

$$\sum_{t \in W_\rho} \Pr[\texttt{Fst}(\rho, t) \text{ or } \texttt{Snd}(\rho, t) \mid p_\rho \in A_t]$$
$$\cdot \Pr[p_\rho \in A_t]$$
$$\leq \sum_{t \in W_\rho^b} \Pr[\texttt{Fst}(\rho, t) \text{ or } \texttt{Snd}(\rho, t) \mid p_\rho \in A_t]$$

(3.13)
$$\cdot \Pr[p_\rho \in A_t]$$
$$+ \sum_{t \in W_\rho^m} \Pr[\texttt{Fst}(\rho, t) \text{ or } \texttt{Snd}(\rho, t) \mid p_\rho \in A_t]$$

(3.14)
$$\cdot \Pr[p_\rho \in A_t]$$

By taking similar steps we did in showing (3.8), we can show that $(3.13) \geq (1 - \epsilon)^2 (1 - 1/2e) z_\rho^b$ and $(3.14) \geq (1 - \epsilon)^2 z_\rho^m$. Hence we obtain

LEMMA 3.6. *Suppose $H \geq 1/\epsilon^3$ and $0 < \epsilon \leq 1/100$. Then any large-window request $\rho \in \mathcal{R}^{large}$ is satisfied (in the first or second round) with a probability of at least $(1 - \epsilon)^2 \big((1 - 1/2e) z_\rho^b + z_\rho^m \big)$.*

We use the same bound shown in Lemma 3.1 for small-window requests, and by summing over all request ρ, we complete the proof of the profit claimed in (3.10).

3.4 Improving the Run Time In this section, we show how to improve the run time of our algorithm to $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$. We first observe that we can easily remove the dependency on the time horizon length T. This can be done by taking similar steps as we took in Section 2.3. Here we explain an easy way of doing this: If there is an interval of length greater than m on which no request's window starts or ends, we throw out all times steps in the interval but m time steps. Note that one can obtain as much profit in in this "down-sized" instance as in the original instance. Also it is easy to observe that the number of remaining time steps is at most $O(m^2)$.

The main idea for reducing the run time from $m^{O(1/\epsilon^4)}$ to $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$ is to solve the $\mathsf{LP_{Throughput}}$ more efficiently. More concretely, we will solve the dual of the LP using an efficient separation oracle. The dual LP is defined as follows.

$(\mathsf{LP_{Dual:Throughput}})$

$$\min \quad \sum_{I \in \mathcal{I}} \gamma_I + \sum_{\rho \in \mathcal{R}^{large}} \xi_\rho$$

$$s.t. \quad w_{I,Q} + \sum_{\rho \in \mathcal{R}^{large} : p_\rho \in P(Q, I \cap W_\rho)} \delta_\rho \leq \gamma_I$$

(3.15)
$$\forall I \in \mathcal{I}, Q \in \mathcal{Q}(I)$$

(3.16)
$$-\delta_\rho + \xi_\rho \leq w_\rho \qquad \forall \rho \in \mathcal{R}^{large}$$

$$\gamma_I \geq 0 \qquad \forall I \in \mathcal{I}$$
$$\delta_\rho, \xi_\rho \geq 0 \qquad \forall \rho \in \mathcal{R}^{large}$$

To avoid considering all constraints (3.15), we use an efficient separation oracle. Since there are only polynomially many constraints (3.16) (with no dependence on ϵ), we focus on giving a separation oracle for constraints (3.15). We observe that for each fixed $I \in \mathcal{I}$, the problem of finding a (if any) violated constraint (3.15) is essentially equivalent to finding a schedule during I that maximizes the throughput when each small-window request ρ has a profit w_ρ and each large-window request ρ has a profit δ_ρ. Here we assume that I are the only time steps that exist (each request ρ's window is restricted to I). Obviously we will need to transmit at most $2H/\epsilon$ pages (the interval I has a length of at most $2H/\epsilon$). Hence at each time step t, as a potential page to transmit at the time, we only need to consider the $2H/\epsilon$ pages that yield the maximum profit assuming that the time step t is the only time step when we transmit a page. Hence we can find the best schedule that maximizes the total profit in I in time $(H/2\epsilon)^{H/2\epsilon} \cdot poly(m) = (1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$. If the configuration corresponding to this schedule violates the constraint (3.15), then we report it as a violated constraint. Otherwise, we conclude that no constraint is violated for the interval $I \in \mathcal{I}$.

We have shown that one can solve the dual LP in $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$ time. This implies that we can find an optimal solution for the dual LP by considering only $(1/\epsilon)^{O(1/\epsilon)} \cdot poly(m)$ many dual constraints. Hence by solving $\mathsf{LP_{Throughput}}$ restricted only to the variables corresponding to those dual constraints, we can obtain an optimal solution for the primal $\mathsf{LP_{Throughput}}$ by the strong duality theorem. The run time of our randomized algorithms is negligible.

References

[1] Airmedia website. http://www.airmedia.com.

[2] Directpc website. http://www.direcpc.com.

[3] Intel intercast website. http://www.iintercast.com.

[4] Open problems in scheduling, 10.02. - 14.02.2010. In Susanne Albers, Sanjoy K. Baruah, Rolf H. Mhring, and Kirk Pruhs, editors, *Scheduling*. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2010 (Can be also found in http://www.cs.pitt.edu/ kirk/dagstuhl2010openproblems.pdf).

[5] S. Acharya, M. Franklin, and S. Zdonik. Dissemination-based data delivery using broadcast disks. *IEEE Pers. Commun.*, 2(6):50–60, Dec 1995.

[6] Demet Aksoy and Michael J. Franklin. ”rxw: A scheduling approach for large-scale on-demand data broadcast. *IEEE/ACM Trans. Netw.*, 7(6):846–860, 1999.

[7] Nikhil Bansal, Moses Charikar, Ravishankar Krishnaswamy, and Shi Li. Better algorithms and hardness for broadcast scheduling via a discrepancy approach. In *SODA*, pages 55–71, 2014.

[8] Nikhil Bansal, Don Coppersmith, and Maxim Sviridenko. Improved approximation algorithms for broadcast scheduling. In *SODA*, pages 344–353, 2006.

[9] Nikhil Bansal, Don Coppersmith, and Maxim Sviridenko. Improved approximation algorithms for broadcast scheduling. *SIAM J. Comput.*, 38(3):1157–1174, 2008.

[10] Nikhil Bansal, Ravishankar Krishnaswamy, and Viswanath Nagarajan. Better scalable algorithms for broadcast scheduling. In *ICALP (1)*, pages 324–335, 2010.

[11] Amotz Bar-Noy, Randeep Bhatia, Joseph (Seffi) Naor, and Baruch Schieber. Minimizing service and operation costs of periodic scheduling. *Math. Oper. Res.*, 27(3):518–544, 2002.

[12] Yair Bartal and S. Muthukrishnan. Minimizing maximum response time in scheduling broadcasts. In *SODA*, pages 558–559, 2000.

[13] Wun-Tat Chan, Tak Wah Lam, Hing-Fung Ting, and Prudence W. H. Wong. New results on on-demand broadcasting with deadline via job scheduling with cancellation. In *COCOON*, pages 210–218, 2004.

[14] Jessica Chang, Thomas Erlebach, Renars Gailis, and Samir Khuller. Broadcast scheduling: Algorithms and complexity. *ACM Transactions on Algorithms*, 7(4):47, 2011.

[15] Chandra Chekuri, Avigdor Gal, Sungjin Im, Samir Khuller, Jian Li, Richard Matthew McCutchen, Benjamin Moseley, and Louiqa Raschid. New models and algorithms for throughput maximization in broadcast scheduling - (extended abstract). In *WAOA*, pages 71–82, 2010.

[16] Chandra Chekuri, Sungjin Im, and Benjamin Moseley. Online scheduling to minimize maximum response time and maximum delay factor. *Theory of Computing*, 8(1):165–195, 2012.

[17] Marek Chrobak. Personal Commnication, 2013.

[18] Marek Chrobak, Christoph Dürr, Wojciech Jawor, Lukasz Kowalik, and Maciej Kurowski. A note on scheduling equal-length jobs to maximize throughput. *J. of Scheduling*, 9(1):71–73, 2006.

[19] R. K. Deb. Optimal control of bulk queues with compound poisson arrivals and batch service. *Opsearch.*, 21:227–245, 1984.

[20] R. K. Deb and R. F. Serfozo. Optimal control of batch service queues. *Adv. Appl. Prob.*, 5:340–361, 1973.

[21] Jeff Edmonds, Sungjin Im, and Benjamin Moseley. Online scalable scheduling for the ;k-norms of flow time without conservation of work. In *SODA*, pages 109–119, 2011.

[22] Uriel Feige and Jan Vondrák. The submodular welfare problem with demand queries. *Theory of Computing*, 6(1):247–290, 2010.

[23] Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. Dependent rounding and its applications to approximation algorithms. *J. ACM*, 53(3):324–360, 2006.

[24] Michel X. Goemans, Maurice Queyranne, Andreas S. Schulz, Martin Skutella, and Yaoguang Wang. Single machine scheduling with release dates. *SIAM J. Discrete Math.*, 15(2):165–192, 2002.

[25] Sungjin Im. *Online Scheduling Algorithms for Average Flow Time and its Variants.* PhD thesis, Computer Science, University of Illinois at Urbana-Champaign, 2012.

[26] Sungjin Im and Benjamin Moseley. An online scalable algorithm for average flow time in broadcast scheduling. In *SODA*, pages 1322–1333, 2010.

[27] Bala Kalyanasundaram and Kirk Pruhs. Speed is as powerful as clairvoyance. *J. ACM*, 47(4):617–643, 2000.

[28] Bala Kalyanasundaram, Kirk Pruhs, and Mahendran Velauthapillai. Scheduling broadcasts in wireless networks. *J. Scheduling*, 4(6):339–354, 2000.

[29] Jae-Hoon Kim and Kyung-Yong Chwa. Scheduling broadcasts with deadlines. *Theor. Comput. Sci.*, 325(3):479–488, 2004.

[30] Colin McDiarmid. Concentration. In Michel Habib, Colin McDiarmid, Jorge Ramirez-Alfonsin, and Bruce Reed, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*, volume 16 of *Algorithms and Combinatorics*, pages 195–248. Springer Berlin Heidelberg, 1998.

[31] Benjamin Moseley. *Online Scheduling Algorithms for Broadcasting and General Cost Functions.* PhD thesis, Computer Science, University of Illinois at Urbana-Champaign, 2012.

[32] Prabhakar Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *J. Comput. Syst. Sci.*, 37(2):130–143, 1988.

[33] Martin Skutella and David P. Williamson. A note on the generalized min-sum set cover problem. *Operations Research Letters*, To appear, 2011.

[34] J. Weiss. Optimal control of batch service queues with nonlinear waiting costs. *Modeling and Simulation*, 10:305–309, 1979.

[35] J. Weiss and S. Pliska. Optimal policies for batch service queueing systems. *Opsearch*, 19(1):12–22, 1982.

[36] J. Wong. Broadcast delivery. *Proc. IEEE*, 76(12):1566–1577, 1988.

[37] Feifeng Zheng, Stanley P. Y. Fung, Wun-Tat Chan, Francis Y. L. Chin, Chung Keung Poon, and Prudence W. H. Wong. Improved on-line broadcast scheduling with deadlines. In *COCOON*, pages 320–329, 2006.

A Concentration Inequalities

The following theorem follows from Bernstein inequalities.

THEOREM A.1. ([30]) *Let $X_1, X_2, ..., X_n$ be n independent random variables such that for all $i \in [n]$, $X_i \leq b$. Let $Y = \sum_{i=i}^{n} X_i$, $\mu := \mathbb{E}[Y]$, and $V := Var[Y]$. Then it follow that*

$$\Pr\left[Y - \mu \geq \Delta\right] \leq \exp\left(-\frac{\Delta^2}{2V(1 + (b\Delta/3V))}\right)$$

B Pseudocodes

INPUT: An estimate L of the maximum flow time L^*; $\mathcal{P}$ and $\mathcal{R}$.

1: **for** $t = L$ to T **do**
2:   $\mathcal{Q}^f(t) \leftarrow \emptyset$.
3: **end for**
4: $\mathcal{Q}^f(L - 1) := \{\{(0, p_0), (1, p_1), ..., (L - 1, p_{L-1})\} \mid p_0, p_1, ..., p_{L-1} \in \mathcal{P}\}$.
5: **for** $t = L$ to T **do**
6:   **for** each $Q = \{(t - L, p_{t-L}), (t - L + 1, p_{t-L+1}), ..., (t - 1, p_{t-1})\} \in \mathcal{Q}^f(t - 1)$ **do**
7:     **for** each $p_t \in \mathcal{P}$ **do**
8:       **if** all requests with release time $t - L$ are satisfied by $p_{t-L+1}, p_{t-L+2}, ..., p_{t-1}$ or p_t **then**
9:         Let $Q' = \{(t - L + 1, p_{t-L+1}), (t - L + 2, p_{t-L+2}), ..., (t, p_t)\}$.
10:        $\text{Prev}(Q') = Q$.
11:        Add Q' to $\mathcal{Q}^f(t)$.
12:      **end if**
13:    **end for**
14: **end for**
15: **end for**
16: **if** $\mathcal{Q}^f(T) = \emptyset$ **then**
17:   Declare that the maximum flow time is greater than L.
18: **else**
19:   Consider any $Q \in \mathcal{Q}^f(T)$.
20:   **for** $t = T$ to L **do**
21:     Let $(t, p_t) \in Q$.
22:     $p'_t \leftarrow p_t$.
23:     $Q \leftarrow \text{Prev}(Q)$.
24:   **end for**
25:   Let $Q = \{(0, p_0), (1, p_1), ..., (L - 1, p_{L-1})\}$.
26:   $p'_1 \leftarrow p_1, p'_2 \leftarrow p_2, ..., p'_L \leftarrow p_L$.
27:   Transmit page p'_t at time t, $1 \leq t \leq T$.
28: **end if**

Algorithm 1: Dynamic Programming with run time $O(T) \cdot n^{O(L^*)}$

INPUT: Modified instance J' with $\ell = 2 + 1/\epsilon$.

1: $T' := \lceil \frac{\max_\rho r_\rho}{\epsilon L^*} \rceil + \ell$.
2: **for** $t = \ell$ to T' **do**
3:   $\mathcal{Q}^f(t) \leftarrow \emptyset$.
4: **end for**
5: $\mathcal{Q}^f(\ell - 1) := \{\{(0, \mathcal{A}_0), (1, \mathcal{A}_1), ..., (\ell - 1, \mathcal{A}_{\ell-1})\} \mid \mathcal{A}_i \in \binom{\cup_{k=0}^{\ell-2} \mathcal{P}_k}{\epsilon L^*}, 0 \leq i \leq \ell - 1\}$.
6: **for** $t = \ell$ to T' **do**
7:   **for** each $Q = \{(t - \ell, \mathcal{A}_{t-\ell}), (t - \ell + 1, \mathcal{A}_{t-\ell+1}), ..., (t - 1, \mathcal{A}_{t-1})\} \in \mathcal{Q}^f(t - 1)$ **do**
8:     **for** each $\mathcal{A}_t \in \binom{\cup_{i=1}^{\ell} \mathcal{P}_{t-i}}{\epsilon L^*}$ **do**
9:       **if** $\mathcal{P}_{t-\ell} \subseteq \bigcup_{i=0}^{\ell-1} \mathcal{A}_{t-i}$ **then**
10:        **if** no page appears more than twice in $\mathcal{A}_{t-\ell+1}, \mathcal{A}_{t-\ell+2}, ..., \mathcal{A}_t$ **then**
11:          Let $Q' = \{(t - \ell + 1, \mathcal{A}_{t-\ell+1}), (t - \ell + 2, \mathcal{A}_{t-\ell+2}), ..., (t, \mathcal{A}_t)\}$.
12:          $\text{Prev}(Q') = Q$.
13:          Add Q' to $\mathcal{Q}^f(t)$.
14:        **end if**
15:      **end if**
16:    **end for**
17: **end for**
18: **end for**
19: **if** $\mathcal{Q}^f(T') = \emptyset$ **then**
20:   Declare that the maximum flow time is greater than ℓ.
21: **else**
22:   Consider any $Q \in \mathcal{Q}^f(T)$.
23:   **for** $t = T'$ to ℓ **do**
24:     Let $(t, \mathcal{A}_t) \in Q$.
25:     $\mathcal{A}'_t \leftarrow \mathcal{A}_t$.
26:     $Q \leftarrow \text{Prev}(Q)$.
27:   **end for**
28:   Let $Q = \{(0, \mathcal{A}_0), (1, \mathcal{A}_1), ..., (\ell - 1, \mathcal{A}_{\ell-1})\}$.
29:   $\mathcal{A}'_1 = \mathcal{A}_1, \mathcal{A}'_2 = \mathcal{A}_2, ..., \mathcal{A}'_{\ell-1} = \mathcal{A}_{\ell-1}$.
30:   Transmit pages $\mathcal{A}'_t$ at time t, $1 \leq t \leq T'$.
31: **end if**

Algorithm 2: Dynamic Programming when $L^* \leq (1/\epsilon^2) \log T$