Algorithmica (2019) 81:1–25
https://doi.org/10.1007/s00453-018-0428-3

New Online Algorithms for Story Scheduling in Web Advertising

Susanne Albers[1] · **Achim Passen**[2]

Received: 13 December 2016 / Accepted: 15 March 2018 / Published online: 22 March 2018

Abstract We study *storyboarding* where advertisers wish to present sequences of ads (stories) uninterruptedly on a major ad position of a web page. These jobs/stories arrive online and are triggered by the browsing history of a user who at any time continues surfing with probability β. The goal of an ad server is to construct a schedule maximizing the expected reward. The problem was introduced by Dasgupta, Ghosh, Nazerzadeh and Raghavan (SODA'09) who presented a 7-competitive online algorithm. They also showed that no deterministic online strategy can achieve a competitiveness smaller than 2, for general β. We present improved algorithms for storyboarding. First we give a simple online strategy that achieves a competitive ratio of $4/(2-\beta)$, which is upper bounded by 4 for any β. The algorithm is also $1/(1-\beta)$-competitive, which gives better bounds for small β. As the main result of this paper we devise a refined algorithm that attains a competitive ratio of $c = 1 + \phi$, where $\phi = (1 + \sqrt{5})/2$ is the Golden Ratio. This performance guarantee of $c \approx 2.618$ is close to the lower bound of 2. Additionally, we study for the first time a problem

A preliminary version of this paper has appeared in *Proc. 40th International Colloquium on Automata, Languages, and Programming (ICALP), 2013*.

S. Albers was supported in part by the European Research Council, Grant Agreement No. 691672.

✉ Susanne Albers
albers@in.tum.de

Achim Passen
passen@informatik.hu-berlin.de

[1] Department of Computer Science, Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany

[2] Department of Computer Science, Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany

extension where stories may be presented simultaneously on several ad positions of a web page. For this parallel setting we provide an algorithm whose competitive ratio is upper bounded by $1/(3-2\sqrt{2}) \approx 5.828$, for any β. All our algorithms work in phases and have to make scheduling decisions only every once in a while.

Keywords Storyboarding · Competitive analysis · One ad position · Multiple ad positions

1 Introduction

Online advertising has grown steadily over the last years. The worldwide online ad spending is expected to reach $160 billion by the end of 2016 [1]. In the US the expenditure for online ads has surpassed that for print ads during the past years and is likely to exceed that for TV ads in 2017 [2]. In this paper we study an algorithmic problem in advertising introduced by Dasgupta et al. [3]. The problem is motivated by *storyboarding*, an advanced online ad format that was first launched by New York Times Digital. It is also referred to as surround sessions [4] and active on many websites these days. In storyboarding, while a user surfs the web and visits a particular website, a single advertiser controls a major ad position for a certain continuous period of time. The advertiser can use these time slots to showcase a range of products and build a linear story line. Typically several advertisers compete for the ad position, depending on the user's browsing history and current actions. The goal of an ad server is to allocate advertisers to the time slots of a user's browsing session so as to maximize the total revenue. Storyboarding is an interesting form of display advertising whose market share is not as high as that of search ads. Nonetheless it has the advantage of strenghtening an advertiser's message. Scientifically it leads to challenging optimization problems.

Dasgupta et al. [3] formulated storyboarding as an online job scheduling problem. Consider a user that starts a web session at time $t = 0$. Time is slotted. At any time t the user continues surfing with probability β, where $0 < \beta \leq 1$, and stops surfing with probability $1 - \beta$. Hence the surfing time is a geometrically distributed random variable. Over time jobs (advertisers) arrive online. These jobs arise based on the user's browsing history and accesses to web content. Each job i is specified by an arrival time a_i, a length l_i and a per-unit value v_i. Here l_i is the length of the ad sequence the advertiser would like to present and v_i is the reward obtained by the server in showing one unit of job i. This reward has to be discounted by the time when the job unit is shown, as specified in the next paragraph. Considering all incoming jobs, we obtain a problem instance $\mathcal{I} = (a_i, v_i, l_i)_{i=1}^{N}$, where $N \in \mathbb{N} \cup \{\infty\}$. We allow $N = \infty$ to model potentially infinitely long browsing sessions and associated job arrivals.

A schedule S for $\mathcal{I}$ specifies which job to process at any time $t \geq 0$. The schedule does not have to contain all jobs; it is allowed to leave out (unattractive) jobs. Schedule S is feasible if every scheduled job i is processed at times $t \geq a_i$ for up to l_i time units. Moreover, it is required that each scheduled job is processed *continuously without* interruption so that an advertiser can build a story. *Preemption* of jobs is allowed, i.e., a job i may be processed for less than l_i time units. In this case no value can be attained for the preempted unscheduled portion of a job. Given a schedule S, its *value*

is defined as the expected value $\sum_{t=0}^{\infty} \beta^t v(t)$, where $v(t)$ is the per-unit value of the job scheduled at time t. The goal is to maximize this reward. Let *ALG* be an online algorithm that, given any input $\mathcal{I}$, constructs a schedule of value $ALG(\mathcal{I})$. Let $OPT(\mathcal{I})$ be the value of an optimal offline schedule for $\mathcal{I}$. Algorithm *ALG* is *c-competitive* if there exists a constant α such that $c \cdot ALG(\mathcal{I}) + \alpha \geq OPT(\mathcal{I})$ holds for all $\mathcal{I}$, cf. [5].

We remark that the online scenario is the setting relevant in practice. As a user visits a website and accesses content, it becomes attractive to several advertisers who then wish to show ad sequences/jobs. In fact, if all jobs (advertisers) were available at time 0, the scheduling problem would be trivial: Simply sequence the jobs in order of non-increasing per-unit value.

1.1 Previous Work

Algorithmic problems in online advertising have received considerable research interest lately, see e.g. [6–14] and references therein. To the best of our knowledge storyboarding, from an algorithmic perspective, has only been studied so far by Dasgupta et al. [3]. A first observation is that if $\beta = 1$, then the scheduling problem is again simple to solve. Every schedule that never preempts jobs and sequences them in an arbitrary order, subject to arrival constraints, achieves an optimal value. Therefore we concentrate on the case that the discount factor β satisfies $0 < \beta < 1$.

Dasgupta et al. [3] showed that no deterministic online algorithm can achieve a competitive ratio smaller than $\beta + \beta^2$. This ratio can be arbitrarily close to 2 as $\beta \to 1$. Hence, for general β, no deterministic online strategy can achieve a competitiveness smaller than 2. As a main result Dasgupta et al. devised a greedy algorithm that is 7-competitive. At any time the algorithm checks if it is worthwhile to preempt the job i currently being executed. To this end the strategy compares the reward obtained in scheduling another unit of job i to the loss incurred in delaying jobs of per-unit value higher than v_i for one time unit.

Furthermore, Dasgupta et al. addressed a problem extension where jobs have increasing rather than constant per-unit values. They focused on the case that value is obtained only when a job is completely finished. The authors showed that no algorithm can achieve a constant competitive ratio and gave a strategy with a logarithmic competitiveness. Finally Dasgupta et al. studied an extension where a job must be scheduled immediately upon arrival; otherwise it is lost. Here they proved a logarithmic lower bound on the performance of any randomized online strategy.

1.2 Our Contribution

We present new and improved online algorithms for storyboarding. All strategies follow the paradigm of processing a given job sequence $\mathcal{I}$ in phases, where a phase consists of k consecutive time steps in the scheduling horizon, for some $k \in \mathbb{N}$. At the beginning of each phase an algorithm computes a schedule for the phase, ignoring jobs that may arrive during the phase. Hence the strategies have to make scheduling decisions only every once in a while.

First in Sect. 2 we give a simple algorithm that computes an optimal schedule for each phase and preempts jobs that are not finished at the end of the respective phase. We prove that the competitive ratio of this strategy is exactly $1/(\beta^{k-1}(1-\beta^k))$, for all $k \in \mathbb{N}$ and all β. The best choice of k gives a competitiveness of $4/(2-\beta)$, which is upper bounded by 4 for any β. If k is set to 1, the resulting algorithm is $1/(1-\beta)$-competitive. This gives further improved bounds for small β, i.e., when $\beta < 2/3$.

In Sect. 3, as our main contribution, we devise a refined algorithm that prefers not to preempt jobs sequenced last in a phase but rather tries to continue them in the following phase. The competitive ratio of this strategy is upper bounded by $1/\beta^{k-1} \cdot \max\{1/\beta^{k-1}, 1/(1-\beta^{2k}), 1+\beta^{3k}/(1-\beta^k)\}$. Using the best choice of k, we obtain a competitive factor of $c = 1+\phi$, where $\phi = (1+\sqrt{5})/2$ is the Golden Ratio. Hence $c \approx 2.618$ and this performance guarantee is close to the lower bound of 2 presented by Dasgupta et al. [3] for general β.

In Sect. 4 we consider for the first time a problem extension where a web page features not only one but several ad positions where stories can be presented simultaneously. This is a natural extension because many web pages do contain a (small) number of ad positions. Again a job sequence $\mathcal{I} = (a_i, v_i, l_i)_{i=1}^N$ is triggered by the browsing history of a user. We assume that an ad server may assign these jobs to a general number m of ad positions. Following the scheduling terminology we refer to these ad positions as machines. In a feasible schedule each job must be processed continuously without interruption on one machine. A migration of jobs among machines is not allowed. The value of a schedule is $\sum_{t=0}^{\infty}\sum_{j=1}^{m}\beta^t v(t,j)$, where $v(t,j)$ is the per-unit value of the job scheduled on machine j at time t. We extend our first algorithm to this parallel setting and derive a strategy that achieves a competitive ratio of $(1+1/(1-\beta(2-\sqrt{2}))/(2-\sqrt{2})$. For small β, this ratio can be as low as $2/(2-\sqrt{2}) \approx 3.414$. For any β, the ratio is upper bounded by $1/(3-2\sqrt{2}) \approx 5.828$.

Technically, in the analyses of the algorithms, we consider quantized inputs in which job arrival times are integer multiples of k. For the setting where one ad position is available (Sects. 2, 3), we are able to prove an interesting property given any quantized input: In an online schedule or a slight modification thereof, no job starts later than in an optimal offline schedule. This property has the important consequence that, for its scheduled job portions, an online algorithm achieves a total value that is at least as high as that of an optimal schedule. Hence the competitive analyses reduce to bounding the loss incurred by an online strategy in preempting jobs. For the refined algorithm this loss analysis is quite involved and in order to prove a small competitive ratio we have to amortize the loss of a preempted job over several phases. In the setting were multiple ad positions are available (Sect. 4), such a property on job starting times does not hold. Therefore we construct a specific optimal schedule $S^\star$ that allows us to match job units sequenced in $S^\star$ to job units sequenced online. Using this matching we can upper bound the additional value achieved by an optimal solution.

2 A 4-Competitive Algorithm

As mentioned before, all algorithms that we present in this paper process a job sequence in phases. Let $k \geq 1$ be an integer. A *k-phase* consists of k consecutive time steps in the

Algorithm $ALG1_k$: Each phase P_n is processed as follows.
Schedule the jobs of Q_n in order of non-increasing per-unit value in P_n. Preempt the last job assigned to P_n if it does not complete by the end of the phase. Execute this schedule for P_n, ignoring jobs that arrive during the phase.

Fig. 1 The algorithm $ALG1_k$

scheduling horizon. More specifically, the n-th k-phase P_n is the subsequence of time steps $(n-1)k, \ldots, nk-1$, for any $n \geq 1$. We remark that $P_n = (n-1)k, \ldots, nk-1$ is a sequence of steps. Our first algorithm, called $ALG1_k$, computes an optimal schedule for any phase, given the jobs that are available at the beginning of the phase. Such an optimal schedule is obtained by simply sequencing the available jobs in order of non-increasing per-unit value. Jobs that arrive during the phase are deferred until the beginning of the next phase.

Formally, $ALG1_k$ works as follow. We say that a job i is *available at time* t if the job has arrived by time t, i.e. $a_i \leq t$, and has not been scheduled so far at any time $t' < t$. Consider an arbitrary phase P_n and let Q_n be the set of jobs that are available at the beginning of P_n. We note that Q_n includes the jobs that arrive at time $(n-1)k$. $ALG1_k$ constructs a schedule for P_n by first sorting the jobs of Q_n in order of non-increasing per-unit value. Jobs having the same per-unit value are sorted in order of increasing arrival times; ties may be broken arbitrarily. Given this sorted sequence, $ALG1_k$ then assigns the jobs one by one to P_n until the k time steps are scheduled or the job sequence ends. In the former case, the last job assigned to P_n is preempted at the end of the phase unless the job completes by the end of P_n. $ALG1_k$ executes this schedule for P_n, ignoring jobs that may arrive during the phase at times $t = (n-1)k+1, \ldots, nk-1$. A summary of $ALG1_k$ is given in Fig. 1.

We first evaluate the performance of $ALG1_k$, for general k. Then we will determine the best choice of k.

Theorem 1 *For all $k \in \mathbb{N}$ and all probabilities β, $ALG1_k$ is $1/(\beta^{k-1}(1-\beta^k))$-competitive.*

In the following we prove the above theorem. Let $\mathcal{I} = (a_i, v_i, l_i)_{i=1}^N$ be an arbitrary input. In processing $\mathcal{I}$, $ALG1_k$ defers jobs arriving after the beginning of a phase until the start of the next phase. Consider a *k-quantized input* $\mathcal{I}_k$ in which the arrival time of any job is set to the next integer multiple of k, i.e. $\mathcal{I}_k = (a_i', v_i, l_i)_{i=1}^N$, where $a_i' = k\lceil a_i/k \rceil$. If a_i is a multiple of k and hence coincides with the beginning of a k-phase, the job is not delayed. Otherwise the job is delayed until the beginning of the next phase. The schedule generated by $ALG1_k$ for $\mathcal{I}_k$ is identical to that computed by $ALG1_k$ for $\mathcal{I}$. Thus $ALG1_k(\mathcal{I}_k) = ALG1_k(\mathcal{I})$. In order to prove Theorem 1 it will be convenient to compare $ALG1_k(\mathcal{I}_k)$ to $OPT(\mathcal{I}_k)$. The next lemma ensures that $OPT(\mathcal{I}_k)$ and the true optimum $OPT(\mathcal{I})$ differ by a factor of at most $1/\beta^{k-1}$.

Lemma 1 *For all $k \in \mathbb{N}$ and all probabilities β, inequality $1/\beta^{k-1} \cdot OPT(\mathcal{I}_k) \geq OPT(\mathcal{I})$ holds.*

Proof Consider an optimal schedule for $\mathcal{I}$ and shift the entire schedule by $k-1$ time units to the right, i.e., the starting time of any job is delayed by exactly $k-1$ time

units. The modified schedule is feasible for $\mathcal{I}_k$ because, for any job i, its arrival time a_i' in $\mathcal{I}_k$ is at most $k-1$ time units later than its arrival a_i in $\mathcal{I}$. The modified schedule has a value of $\beta^{k-1}OPT(\mathcal{I})$, and an optimal schedule for $\mathcal{I}_k$ achieves a value at least that high. □

In order to estimate $OPT(\mathcal{I}_k)$ we consider a stronger optimal offline algorithm that was also proposed by Dasgupta et al. [3]. This algorithm is allowed to resume interrupted jobs at a later point in time. We call this offline strategy *CHOP*. For any input, at any time t *CHOP* schedules a job having the highest per-unit value among the unfinished jobs that have arrived until time t. Obviously, $CHOP(\mathcal{I}_k) \geq OPT(\mathcal{I}_k)$. Let S be the schedule computed by $ALG1_k$ for $\mathcal{I}_k$ and let $S^\star$ be the schedule generated by *CHOP* for $\mathcal{I}_k$. We assume w.l.o.g. that in $S^\star$ all jobs having a certain per-unit value v are processed in the same order as in S. More specifically, all jobs having per-unit value v are processed in order of increasing arrival times. Jobs of per-unit value v arriving at the same time are processed in the same order as in S. Schedule $S^\star$ can be easily modified so that this property is satisfied. For any job i, let $t_S(i)$ denote its starting time in S and let $t_{S^\star}(i)$ be its starting time in $S^\star$. If job i is never processed in S (or $S^\star$), then we set $t_S(i) = \infty$ (or $t_{S^\star}(i) = \infty$). The following lemma states that $ALG1_k$ starts each job at least as early as *CHOP*.

Lemma 2 *For any job i, $t_S(i) \leq t_{S^\star}(i)$.*

Proof The desired inequality obviously holds for jobs never scheduled by *CHOP*. Suppose that the lemma does not hold for every job and let i be the one occuring earliest in $S^\star$ with $t_{S^\star}(i) < t_S(i)$. Let $t^\star = t_{S^\star}(i)$ and P_n be the phase containing $t^\star$. Moreover, let job j be the one scheduled by $ALG1_k$ at time $t^\star$. In $\mathcal{I}_k$ jobs arrive only at the beginning of a phase when $ALG1_k$ makes scheduling decisions. Hence at the beginning of P_n job i has arrived and can be scheduled by $ALG1_k$. Since $ALG1_k$ sequences available jobs in order of non-increasing per-unit value and does not start job i at or before time $t^\star$, there holds $v_j \geq v_i$.

We next argue that at time $t^\star$ *CHOP* has already finished job j. This clearly holds if $v_j > v_i$ because *CHOP* always schedules an unfinished job with the highest per-unit value. If $v_j = v_i$, then again *CHOP* must have completed job j because in $S^\star$ jobs of per-unit value $v = v_j = v_i$ occur in the same order as in S and job j precedes job i in S.

Since *CHOP* has finished job j, it started this job at or before time $t^\star - l_j$. On the other hand $ALG1_k$ did not start job j before time $t^\star - l_j + 1$ because it is still processing this job. We conclude $t_{S^\star}(j) < t_S(j) \leq t_{S^\star}(i)$, which contradicts the assumption that job i is the first one in $S^\star$ violating the desired inequality. □

The next lemma relates the value of $ALG1_k$ to that of OPT, attained for input $\mathcal{I}_k$.

Lemma 3 *For all $k \in \mathbb{N}$ and all probabilities β, inequality $1/(1-\beta^k) \cdot ALG1_k(\mathcal{I}_k) \geq OPT(\mathcal{I}_k)$ holds.*

Proof For any $n \geq 1$, let I_n be the set of jobs scheduled by $ALG1_k$ in phase P_n, i.e., formally
$I_n = \{i \mid (n-1)k \leq t_S(i) \leq nk-1\}$. Let $ALG1_k(P_n)$ be the value achieved by

$ALG1_k$ in scheduling the jobs of I_n, and let $CHOP(P_n)$ be the value achieved by $CHOP$ in processing these jobs. There holds $ALG1_k(\mathcal{I}_k) = \sum_n ALG1_k(P_n)$. A consequence of Lemma 2 is that all jobs that are ever scheduled by $CHOP$ also occur in $ALG1_k$'s schedule. Hence $CHOP(\mathcal{I}_k) = \sum_n CHOP(P_n)$. We will show that, for every $n \in \mathbb{N}$, inequality $CHOP(P_n)/ALG1_k(P_n) \leq 1/(1-\beta^k)$ holds. This implies $CHOP(\mathcal{I}_k)/ALG1_k(\mathcal{I}_k) \leq 1/(1-\beta^k)$ and the lemma then follows because $CHOP(\mathcal{I}_k) \geq OPT(\mathcal{I}_k)$.

Consider any k-phase P_n. In the schedule S let j be the last job started in P_n and let λ_j be the number of time units for which j is sequenced in P_n and thus in the entire schedule S. By Lemma 2, for any job i, there holds $t_S(i) \leq t_{S^\star}(i)$. Hence the total value achieved by $CHOP$ in scheduling the jobs $i \in I_n$ with $i \neq j$ as well as the first λ_j time units of job j cannot be higher than $ALG1_k(P_n)$.

If job j is preempted in S at the end of P_n, then $CHOP$ can achieve an additional value in scheduling units $\lambda_j + 1, \ldots, l_j$ of job j in $S^\star$. Again, since $t_S(j) \leq t_{S^\star}(j)$, these units cannot be sequenced before the beginning of phase P_{n+1}, i.e., at time nk. Thus the additional value achievable for units $\lambda_j + 1, \ldots, l_j$ is upper bounded by $\sum_{t=nk}^{\infty} \beta^t v_j = \beta^{nk}/(1-\beta) \cdot v_j$, which is obtained if a job of per-unit value v_j and infinite length is sequenced starting at time nk.

Thus $CHOP(P_n) \leq ALG1_k(P_n) + \beta^{nk}/(1-\beta) \cdot v_j$. In each phase $ALG1_k$ sequences jobs in order of non-increasing per-unit value. Hence each job of I_n has a per-unit value of at least v_j. We conclude $ALG1_k(P_n) \geq \sum_{t=(n-1)k}^{nk-1} \beta^t v_j = (\beta^{(n-1)k} - \beta^{nk})/(1-\beta) \cdot v_j$ and $CHOP(P_n)/ALG1_k(P_n) \leq 1 + \beta^{nk}/(\beta^{(n-1)k} - \beta^{nk}) = 1/(1-\beta^k)$. □

We are ready to prove Theorem 1.

Proof of Theorem 1 Combining Lemmas 1 and 3 we obtain that, for all $k \in \mathbb{N}$ and all probabilities β, inequality $1/(\beta^{k-1}(1-\beta^k))ALG1_k(\mathcal{I}_k) \geq OPT(\mathcal{I})$ holds for any input $\mathcal{I}$. Since $ALG1_k(\mathcal{I}) = ALG1_k(\mathcal{I}_k)$ the theorem follows. □

We determine the best value of k.

Corollary 1 *For $k = \lceil -\log_\beta 2 \rceil$, the resulting algorithm $ALG1_k$ is $4/(2-\beta)$-competitive.*

Proof The function $f(x) = \beta^{x-1}(1-\beta^x)$ is maximized for $x^\star := -\log_\beta 2$. Choosing $k = \lceil -\log_\beta 2 \rceil$ gives $x^\star \leq k \leq x^\star + 1$ and $f(x^\star) \geq f(k) \geq (2-\beta)/4 = f(x^\star + 1)$ because $f(x)$ is strictly decreasing for $x > x^\star$. Since $ALG1_k$'s competitiveness is $1/f(k)$, the corollary follows. □

The next theorem shows that our analysis of $ALG1_k$ is tight.

Theorem 2 *For all $k \in \mathbb{N}$ and all probabilities β, the competitive ratio of $ALG1_k$ is not smaller than $1/(\beta^{k-1}(1-\beta^k))$.*

Proof Suppose that $ALG1_k$ achieved a competitive ratio $c < 1/(\beta^{k-1}(1-\beta^k))$. Then there exists a constant α such that $c \cdot ALG1_k(\mathcal{I}) + \alpha \geq OPT(\mathcal{I})$ holds for all inputs $\mathcal{I}$. Consider the specific input $\mathcal{I}$ consisting of a single job that arrives at time 1, has a value of $v = \alpha/(\beta(1 - c\beta^{k-1}(1-\beta^k))$, and infinite length. $ALG1_k$ starts this job at

time k and processes it for k time units so that $ALG1_k(\mathcal{I}) = \beta^k(1-\beta^k)/(1-\beta)\cdot v$. On the other hand $OPT(\mathcal{I}) = \beta/(1-\beta)\cdot v$. Hence

$$\begin{aligned} c\cdot ALG1_k(\mathcal{I}) + \alpha &= c\cdot ALG1_k(\mathcal{I}) + (\alpha/v)v = c\cdot ALG1_k(\mathcal{I}) + \beta(1-c\beta^{k-1}(1-\beta^k))v \\ &< c\cdot ALG1_k(\mathcal{I}) + \tfrac{\beta}{1-\beta}(1-c\beta^{k-1}(1-\beta^k))v = \tfrac{\beta}{1-\beta}\cdot v = OPT(\mathcal{I}), \end{aligned}$$

where the inequality holds because $1-\beta < 1$. We obtain a contradiction. □

Finally in this section we consider the algorithm $ALG1_1$ in which the phase length k is set to 1. This algorithm at any time schedules a job having the highest per-unit value among the available jobs. This job is processed for one time unit.

Corollary 2 *For all probabilities β, the competitive ratio of $ALG1_1$ is exactly $1/(1-\beta)$.*

Proof Theorem 1 implies that $ALG1_1$ is $1/(1-\beta)$-competitive. By Theorem 2 the competitive ratio is not smaller than this value. □

We finally combine Corollaries 1 and 2 to obtain the following result.

Corollary 3 *Setting $k = 1$ if $\beta \le 2/3$ and $k = \lceil -\log_\beta 2\rceil$ otherwise, we obtain an algorithm $ALG1_k$ that achieves a competitive ratio of* $\min\{1/(1-\beta), 4/(2-\beta)\}$.

Proof We observe that $1/(1-\beta) \le 4/(2-\beta)$ holds true if and only if $\beta \le 2/3$. Suppose that $\beta \le 2/3$. In this case the resulting algorithm $ALG1_1$ achieves a competitive ratio of $1/(1-\beta) \le \min\{1/(1-\beta), 4/(2-\beta)\}$, cf. Corollary 2. On the other hand assume that $\beta > 2/3$. By Corollary 1, algorithm $ALG1_k$ with $k = \lceil -\log_\beta 2\rceil$ achieves a competitiveness of $4/(2-\beta) < \min\{1/(1-\beta), 4/(2-\beta)\}$. □

3 A Refined Algorithm

We present a second algorithm that, compared to $ALG1_k$, reduces loss incurred in preempting jobs. The algorithm also operates in k-phases. Its crucial property is that it continues processing a job scheduled last in a phase if this job is among the highest-valued jobs available at the beginning of the next phase.

The refined algorithm, called $ALG2_k$, works in two steps. Again, let P_n be any k-phase. Step (1) is defined as follows. If $n > 1$, then let i_n be the job that was scheduled last in P_{n-1} and can potentially be continued in P_n. If this job has been scheduled for less than l_{i_n} time units in the prior schedule, until the end of P_{n-1}, then define a residual job i_n^r by $(a_{i_n}, v_{i_n}, l_{i_n}^r)$. Here $l_{i_n}^r$ is the remaining length of job i_n, i.e., $l_{i_n}^r$ further units have to be processed to complete the job. Let Q_n be the set consisting of job i_n^r and the jobs available at the beginning of P_n. $ALG2_k$ schedules the jobs of Q_n in order of non-increasing per-unit values in P_n. Again, jobs having the same per-unit value are scheduled in order of increasing arrival times, where ties may be broken arbitrarily. Among jobs having a per-unit value of $v = v_{i_n}$, job i_n^r is scheduled first. Let $S'(P_n)$ denote the schedule obtained for P_n at this point.

We next describe Step (2). If $S'(P_n)$ does not contain job i_n^r, then $S'(P_n)$ is equal to the final schedule $S(P_n)$ for the phase. If $S'(P_n)$ contains job i_n^r and this job is

Algorithm $ALG2_k$: Each phase P_n is handled as follows.
(1) If $n > 1$, let i_n be the job scheduled last in P_{n-1}. If job i_n has been scheduled for less than l_{i_n} time units so far, define job i_n^r by $(a_{i_n}, v_{i_n}, l_{i_n}^r)$ and add it to Q_n. Let $S'(P_n)$ be the schedule obtained by sequencing the jobs of Q_n in order of non-increasing per-unit value in P_n.
(2) If $S'(P_n)$ processes job i_n^r for s_n^r time units staring at time t_n^r, then schedule job i_n for s_n^r time units at the beginning of P_n. Jobs originally processed from time $(n-1)k$ to $t_n^r - 1$ are delayed by s_n^r time units. Execute this schedule $S(P_n)$ for P_n, ignoring jobs that arrive during the phase.

Fig. 2 The algorithm $ALG2_k$

scheduled for s_n^r time units starting at time t_n^r in P_n, then $ALG2_k$ modifies $S'(P_n)$ so as to obtain a feasible schedule. Loosely speaking, job i_n^r is shifted to the beginning of P_n. More precisely, the original job i_n is scheduled for s_n^r time units at the beginning of P_n. The start of all jobs scheduled from time $(n-1)k$ to time $t_n^r - 1$ in $S'(P_n)$ is delayed by s_n^r time units. Between time $t_n^r + s_n^r$ and the end of P_n, no modification is required. The resulting schedule is the final output $S(P_n)$. While this schedule is executed, newly arriving jobs are deferred until the beginning of the next phase.

A pseudo-code description of $ALG2_k$ is given in Fig. 2. We remark that a long job i may be executed over several phases, provided that its per-unit value is sufficiently high. In this case the corresponding residual job, defined for a phase P_n, occupies the entire schedule $S'(P_n)$.

Theorem 3 *For all $k \in \mathbb{N}$ and all probabilities β, algorithm $ALG2_k$ achieves a competitive ratio of $1/\beta^{k-1} \cdot \max\{1/\beta^{k-1}, 1/(1-\beta^{2k}), 1+\beta^{3k}/(1-\beta^k)\}$.*

We proceed to prove the above theorem. Compared to the proof of Theorem 1 the analysis is more involved because we have to take care of the delays incurred by $ALG2_k$ in Step (2) when scheduling a portion of job i_n at the beginning of phase P_n and thereby postponing the start of jobs with higher per-unit values. Furthermore, in order to achieve a small competitive ratio we have to charge the loss of a job preempted in a phase to several adjacent phases.

Again, for any input $\mathcal{I} = (a_i, v_i, l_i)_{i=1}^N$, we consider the k-quantized input $\mathcal{I}_k = (a_i', v_i, l_i)_{i=1}^N$, where the arrival time of any job i is set to $a_i' = k\lceil a_i/k \rceil$. There holds $ALG2_k(\mathcal{I}_k) = ALG2_k(\mathcal{I})$ and, as shown in Lemma 1, $1/\beta^{k-1} OPT(\mathcal{I}_k) \geq OPT(\mathcal{I})$. We will compare $ALG2_k(\mathcal{I}_k)$ to $CHOP(\mathcal{I}_k)$, where $CHOP$ is the stronger optimal offline algorithm described in Sect. 2. Again let S denote the schedule computed by $ALG2_k$ for $\mathcal{I}_k$ and let $S^\star$ be $CHOP$'s schedule for $\mathcal{I}_k$. As in Sect. 2 we assume w.l.o.g. that in $S^\star$ jobs having a certain per-unit value v are processed in the same order as in S.

In order to evaluate $ALG2_k(\mathcal{I}_k)$, we define a schedule S' that allows us to prove a statement analogous to Lemma 2 and, moreover, to compare the per-unit values of jobs scheduled in S' and $S^\star$. For any phase P_n, consider the schedule $S'(P_n)$ computed in Step (1) of $ALG2_k$. If $n > 1$ and the residual job i_n^r is scheduled for s_n^r time units starting at time t_n^r in P_n, then modify $S'(P_n)$ by scheduling the *original* job i_n for s_n^r time units starting at time t_n^r. By slightly overloading notation, we refer to this modified schedule as $S'(P_n)$. Schedule S' is the concatenation of the $S'(P_n)$, for all $n \geq 1$.

In $S'(P_n)$ jobs are sequenced in order of non-increasing per-unit value. Among jobs of per-unit value $v = v_{i_n}$, job i_n is processed first. Schedule $S'(P_n)$ differs from $S(P_n)$ only in that job i_n is sequenced after the jobs having a strictly higher per-unit value than v_{i_n}. Each such job starts and finishes in P_n. The shift of the job portion of i_n does not affect the relative order of jobs having the same per-unit value. Hence in S' and S, and thus in S' and $S^\star$, jobs of a certain per-unit value v occur in the same relative order. We note that schedule S' is infeasible in that a job i_n may be interrupted at the end of phase P_{n-1} and resumed later in P_n.

For any job i, let $t_{S'}(i)$ be its starting time in S', i.e., the earliest time when a portion of job i is processed. As usual $t_S(i)$ and $t_{S^\star}(i)$ denote the starting time of job i in S and $S^\star$, respectively. Jobs that never appear in a schedule have a starting time of infinity. In Lemma 5 below we will prove a statement corresponding to that of Lemma 2: For any job i, there holds $t_{S'}(i) \leq t_{S^\star}(i)$. For the proof of this lemma we need the following auxiliary lemma which implies, in particular, that each job is interrupted at most once in S'. The lemma will also be essential in the proof of Lemma 6.

Lemma 4 *If a job is interrupted in S', then this interruption occurs at the end of a phase P_{n-1} and the job is equal to i_n processed last in $S(P_{n-1})$ and $S'(P_{n-1})$. The job is scheduled again only in P_n and experiences no further interruption in S'.*

Proof Consider the schedule S and successively replace $S(P_n)$ by $S'(P_n)$, for increasing phase number n. We identify the interruptions introduced by these replacements. As mentioned above $S'(P_n)$ is equal to $S(P_n)$ except that job i_n, if it occurs in $S(P_n)$, is sequenced after the jobs having a higher per-unit value than v_{i_n}. All of these jobs start and finish in $S(P_n)$ and thus are not interrupted in $S'(P_n)$. Hence when replacing $S(P_n)$ by $S'(P_n)$ only job i_n can get interrupted and in this case job i_n is scheduled last in $S(P_{n-1})$ and $S'(P_{n-1})$. If job i_n is indeed interrupted at the end of $S'(P_{n-1})$ and later continued in $S(P_n)$, then it is not processed throughout the entire phase P_n. Hence in $S(P_n)$ job i_n is not processed until the end of P_n and cannot be processed further in any subsequent phase. Since S' and S sequence the same set of jobs within each phase, the lemma follows. □

Lemma 5 *For any job i, $t_{S'}(i) \leq t_{S^\star}(i)$.*

Proof The proof of the lemma is similar to that of Lemma 2. However, here we have to distinguish cases depending on whether or not job i is interrupted in S'.

For jobs that are never scheduled by *CHOP* there is nothing to show. Suppose that the desired inequality does not hold for all jobs and let job i be the one occuring earliest in $S^\star$ with $t_{S'}(i) > t_{S^\star}(i)$. Set $t^\star = t_{S^\star}(i)$ and let P_n be the phase containing time $t^\star$. Let job j be the one processed in S' at time $t^\star$. There holds $v_j \geq v_i$ because job i has arrived by time $(n-1)t$, the beginning of P_n, and in S' jobs are scheduled in order of non-increasing per-unit value within each phase. We next argue that *CHOP* finishes job j before time $t^\star$. This obviously holds if $v_j > v_i$ because *CHOP* always processes an unfinished job with the highest per-unit value. If $v_j = v_i$, then again *CHOP* must have finished job j because in S' job j occurs before job i and jobs having the same per-unit value are processed in the same order in S' and $S^\star$.

We next distinguish two cases. If job j has not been interrupted in S' before time $t^\star$, we have $t_{S'}(j) \geq t^\star - l_j + 1$. On the other hand, $t_{S^\star}(j) \leq t^\star - l_j$ because *CHOP* has

finished job j before time $t^\star$. We obtain $t_{S^\star}(j) < t_{S'}(j) \leq t^\star$, which is a contradiction to the assumption that job i is the first one in $S^\star$ violating the desired inequality.

If job j has been interrupted in S' before time $t^\star$, then by Lemma 4 it is equal to job i_n processed at the end of P_{n-1}. Let I be the set of jobs that are scheduled between the beginning of P_n and $t^\star$ in S' and are not equal to job j. All of these jobs start and finish in P_n and have a strictly higher per-unit value than job j. Let l be the total length of the jobs in I. By Lemma 4, job j was interrupted only once and hence $t_{S'}(j) \geq t^\star - l_j - l + 1$. Since all jobs of I have a higher per-unit value than v_j and $v_j \geq v_i$, *CHOP* must have finished all of them before time $t^\star$. Therefore $t_{S^\star}(j) \leq t^\star - l_j - l$ because, by assumption, $t_{S'}(i') \leq t_{S^\star}(i')$ holds for all $i' \in I$. We conclude again $t_{S^\star}(j) < t_{S'}(j) \leq t^\star$ and obtain a contradiction to our initial assumption. □

A main goal of the subsequent analysis is to bound the loss incurred by $ALG2_k$ in preempting jobs. The following Lemma 7 will be crucial as it specifies the earliest time when a job preempted in S can occur again in $S^\star$. The proof relies on Lemma 6 below that compares per-unit values of jobs scheduled in S and S'. At any time t, let $v_{S^\star}(t)$ be the per-unit value of the job scheduled in $S^\star$ and let $v_{S'}(t)$ be the per-unit value of the job scheduled in S'. If at time t no job is scheduled in S' or $S^\star$, then the corresponding value $v_{S'}(t)$ or $v_{S^\star}(t)$ is zero.

Lemma 6 *For any time t, $v_{S^\star}(t) \geq v_{S'}(t)$.*

Proof Consider any time t. If no job is scheduled at time t in S', there is nothing to show. Otherwise let job i be the one scheduled in S' at time t. By Lemma 5 there holds $t_{S'}(i) \leq t_{S^\star}(i)$. If job i is not interrupted in S', then $t_{S'}(i) \geq t - l_i + 1$ and hence $t_{S^\star}(i) \geq t - l_i + 1$. Thus *CHOP* does not finish job i before time t and processes a job of per-unit value at least v_i at time t in $S^\star$. If job i is interrupted in S', then let P_n be the phase containing t. By Lemma 4, job i is equal to job i_n processed last in $S'(P_{n-1})$. Let I be the set of jobs $j \neq i$ processed between the beginning of P_n and time t in S'. All these jobs start in $S'(P_n)$. Let l be the total length of the jobs in I. Since job i is interrupted only once, see again Lemma 4, $t_{S'}(i) \geq t - l_i - l + 1$. Using Lemma 5 we obtain $t_{S^\star}(i) \geq t - l_i - l + 1$ and $t_{S'}(j) \leq t_{S^\star}(j)$, for all $j \in I$. This implies that *CHOP* cannot finish all jobs of $I \cup \{i\}$ before time t. By the definition of S', jobs are processed in non-increasing order of per-unit value in each phase. Among jobs of value v_{i_n} job i_n is processed first. Thus all jobs of I have a per-unit-value higher than v_i. We conclude that at time time t *CHOP* processes a job of per-unit value at least v_i. □

Lemma 7 *If job i is preempted in $S(P_n)$ and the following phase schedules $S(P_{n+1}), \ldots, S(P_{n'})$ only process jobs of per-unit value higher than v_i, then $S^\star$ does not schedule job i in phases $P_{n+1}, \ldots, P_{n'}$.*

Proof For any phase P_l, schedules $S(P_l)$ and $S'(P_l)$ process the same set of jobs; of course some jobs might be processed only partially in those schedules. Hence $S'(P_{n+1}), \ldots, S'(P_{n'})$ only process jobs having a per-unit value higher than v_i. By Lemma 6, at any time schedule $S^\star$ processes a job whose per-unit value is at least as high as that of the job scheduled in S'. Hence in phases $P_{n+1}, \ldots, P_{n'}$ schedule $S^\star$ only processes jobs having a per-unit-value higher than v_i. □

In the remainder of the analysis we first classify phases and then compose segments of up to three consecutive phases. For these segments we will upper bound the loss incurred by $ALG2_k$ in preempting jobs.

3.1 Phase Classification

We classify phases, considering the original schedule S. A phase P_n is called *preempted* if a job is preempted in $S(P_n)$. Phase P_n is called *continued* if the job scheduled last in $S(P_n)$ is also scheduled at the beginning of $S(P_{n+1})$. Phase P_n is *complete* if all jobs scheduled in $S(P_n)$ are finished by the end of P_n.

We mention and verify some properties of these phases in the schedule S. (a) In each phase P_n at most one job is preempted in $S(P_n)$. (b) If P_n is a continued or complete phase, no job is preempted in $S(P_n)$. (c) If P_n is a preempted phase, then the job preempted is one having the smallest per-unit value among jobs scheduled in $S(P_n)$. These properties can be verified as follows. Let P_n be an arbitrary phase. When $ALG2_k$ constructs a schedule for P_n, it firsts sorts the jobs of Q_n in order of non-increasing per-unit value. In this sorted sequence only the last job, say job i, assigned to P_n might not be scheduled completely in the phase and hence is a candidate for preemption. Job i is one having the smallest per-unit value among jobs scheduled in the phase. This shows properties (a) and (c). If job i is not moved to the beginning of the phase in Step (2) of $ALG2_k$ and continued at the beginning of the next phase, then P_n is a continued phase and no job is preempted in $S(P_n)$. By definition, no job is preempted in a complete phase. This shows property (b). We observe that in the schedule S each phase is either preempted, continued or complete.

3.2 Schedule Segments

For the further analysis we partition the schedule S into *segments* where a segment consists of up to three consecutive phases. The purpose of these segments is to combine "expensive" preempted phases with other phases so as to amortize preemption loss. First we build segments consisting of three phases. Phases P_n, P_{n+1}, P_{n+2} form a segment if P_n is a preempted phase that is not preceded by a continued phase, P_{n+1} is a continued phase and P_{n+2} is a preempted phase. Among the remaining phases we build segments consisting of two phases. Phases P_n, P_{n+1} form a segment if (a) P_n is a preempted phase that is not preceded by a continued phase and P_{n+1} is a continued or complete phase or (b) P_n is a continued phase followed by a preempted phase P_{n+1}. Each remaining phase forms a separate segment.

We argue that the above segmentation is well defined. First consider the 3-phase segments built initially. No two such segments σ and σ' can overlap: The first phase P'_n of σ' is a preempted phase that is not preceded by a continued phase. Phase P'_n cannot be equal to the second phase of σ because the latter one is continued phase. Furthermore, P'_n cannot be equal to the third phase of σ because that one is a preempted phase that is preceded by a continued phase. Next consider any two 2-phase segments σ and σ' built up. Again no overlap can occur. Suppose that the first phase P'_n of σ' is a preempted phase that is not preceded by a continued phase. P'_n cannot be identical

to the second phase of σ because the latter is a continued phase, a complete phase or a preempted phase that is preceded by a continued phase. Next assume that P'_n is a continued phase that is followed by a preempted phase. If P'_n coincided with the second phase of σ, then the three phases of σ and σ' would form a 3-phase segment built initially because the first phase of σ is a preempted phase that is not preceded by a continued phase.

We next state two properties of a preempted phase P_n that forms a separate 1-phase segment. Firstly, such a phase cannot be preceded by a continued phase: If P_n was preceded by a continued phase P_{n-1}, then P_{n-1} could be the second phase of a 2-phase segement or a separate 1-phase segement. In the former case the 2-phase segment and P_n would form a 3-phase segment built initially. In the latter case P_{n-1} and P_n would form a 2-phase segement as described in property (b) above. A second property is that P_n is followed by a preempted phase. If it was followed by a continued phase P_{n+1}, then P_{n+1} could be the beginning of a 2-phase segment desribed in property (a) above or a separate 1-phase segment. In the former case P_n and the 2-phase segment would combine to a 3-phase segement because P_n is not preceded by a continued phase. In the latter case P_n and P_{n+1} would form a 2-phase segment as desribed in property (a). In summary, a preempted phase that forms a separate 1-phase segment is not preceded by a continued phase and is followed by a preempted phase.

For a segment σ, let $ALG2_k(\sigma)$ be the value achieved by $ALG2_k$ on σ. More specifically, let I be the set of jobs scheduled by $ALG2_k$ in the phases of σ. Set I also includes those jobs that are only partially processed in σ and might also be scheduled in phases before or after σ. Suppose that job $i \in I$ is processed for δ_i time units starting at time t_i is σ. Then

$$ALG2_k(\sigma) = \sum_{i \in I} \beta^{t_i}(1 - \beta^{\delta_i})/(1 - \beta) \cdot v_i.$$

Let $CHOP(\sigma)$ denote the value achieved by $CHOP$ in processing the jobs and job portions scheduled by $ALG2_k$ in σ. More specifically, suppose that in S job $i \in I$ has been processed for λ_i time units before the beginning of σ. Then $CHOP(\sigma)$ represents the value achieved by $CHOP$ in processing the units $\lambda_i + 1, \ldots, \lambda_i + \delta_i$ of job i in $S^\star$. If job i is preempted in the segment σ of S, then $CHOP(\sigma)$ additionally represents the value achieved by processing units $u > \lambda_i + \delta_i$ in $S^\star$. There holds

$$CHOP(\sigma) \leq \sum_{i \in I} \beta^{t_{S^\star}(i)+\lambda_i}(1 - \beta^{\delta_i})/(1 - \beta) \cdot v_i + v_p(\sigma),$$

where $v_p(\sigma)$ denotes the additional value achieved by $CHOP$ for jobs preempted by $ALG2_k$ in σ. There holds $ALG2_k(\mathcal{I}_k) = \sum_\sigma ALG2_k(\sigma)$ and $CHOP(\mathcal{I}_k) = \sum_\sigma CHOP(\sigma)$ because, by Lemma 5, every job scheduled by $CHOP$ is also scheduled by $ALG2_k$.

3.3 Segement Analysis

We prove three lemmas (Lemmas 8, 9 and 10) that upper bound $CHOP(\sigma)/ALG2_k(\sigma)$, for the various segments. Together they will imply Theorem 3. In the proofs of the lemmas we use the following notation. For any phase P_n let I_n denote the set of jobs that are partially or completely processed in $S(P_n)$. For any $i \in I_n$, let $\delta_{i,n}$ be the number of time units for which job i is processed in $S(P_n)$. If job i only scheduled in phase P_n of S, then we simply set $\delta_i = \delta_{i,n}$. Furthermore, let i_n^1 be the first job scheduled in $S(P_n)$. Suppose that P_n is preceded by a continued phase. When constructing $S(P_n)$, $ALG2_k$ might have delayed the starting times of some jobs of I_n in Step (2) in order to move job i_n^1 to the beginning of the phase. Let $I'_n \subset I_n$ be the set of these delayed jobs. If P_n is not preceded by a continued phase, there are no delayed jobs and we set $I'_n = \emptyset$. We observe that $t_S(i) = t_{S'}(i)$, for all $i \in I_n \setminus I'_n \cup \{i_n^1\}$, and $t_S(i) = t_{S'}(i) + \delta_{i_n^1,n}$, for all $i \in I'_n$.

For ease of exposition, let $w(t, \delta, v) = \beta^t(1-\beta^\delta)/(1-\beta)\cdot v$ be the value achieved in processing a job of per-unit value v for δ time units starting at time t. We allow $\delta = \infty$, meaning that a job of infinite length is scheduled starting at time t.

Lemma 8 *For any σ consisting of one phase, $CHOP(\sigma)/ALG2_k(\sigma) \le \max\{1/\beta^{k-1}, 1 + \beta^{3k}/(1-\beta^k)\}$.*

Proof We first study the case that the phase P_n of σ is a continued or complete phase, i.e., no job is preempted in $S(P_n)$. Let $i_1 = i_n^1$ be the first job scheduled in $S(P_n)$. There holds

$$\begin{aligned} ALG2_k(\sigma) &= w((n-1)k, \delta_{i_1,n}, v_{i_1}) + \sum_{i\in I'_n} w(t_{S'}(i) + \delta_{i_1,n}, \delta_{i,n}, v_i) \\ &\quad + \sum_{i\in I_n\setminus(I'_n\cup\{i_1\})} w(t_{S'}(i), \delta_{i,n}, v_i) \\ &\ge \beta^{k-1}\left(w((n-1)k, \delta_{i_1,n}, v_{i_1}) + \sum_{i\in I_n\setminus\{i_1\}} w(t_{S'}(i), \delta_{i,n}, v_i)\right). \end{aligned}$$

The last inequality holds because any job $i \in I'_n$ is delayed by at most $k-1$ time units and hence, for any $i \in I'_n$, we have $w(t_{S'}(i)+\delta_{i_1,n}, \delta_i, v_i) = \beta^{\delta_{i_1,n}} w(t_{S'}(i), \delta_{i,n}, v_i) \ge \beta^{k-1} w(t_{S'}(i), \delta_{i,n}, v_i)$. Every job $i \in I_n$, except for possibly i_1, is started in $S(P_n)$. If job i_1 is started in $S(P_{n'})$, where $n' < n$, then the job is scheduled at the end of $S(P_{n'})$. Hence when $ALG2_k$ constructed $S(P_{n'})$, the job was not delayed in Step (2) of the algorithm in order to move another job to the beginning of the phase. Thus $t_S(i_1) = t_{S'}(i_1) \le t_{S^\star}(i_1)$. Suppose that before P_n job i_1 was processed for λ_{i_1} time units in S. Since $t_S(i_1) \le t_{S^\star}(i_1)$, the units $\lambda_{i_1}+1, \ldots, \lambda_{i_1}+\delta_{i_1,n}$ of job i_1 cannot be started before the beginning of P_n in $S^\star$. For all jobs $i \in I_n \setminus \{i_1\}$, there holds $t_{S'}(i) \le t_{S^\star}(i)$. Hence

$$CHOP(\sigma) \le w((n-1)k, \delta_{i_1,n}, v_{i_1}) + \sum_{i\in I_n\setminus\{i_1\}} w(t_{S'}(i), \delta_{i,n}, v_i).$$

We obtain $CHOP(\sigma)/ALG2_k(\sigma) \leq 1/\beta^{k-1}$.

We next study the case that P_n is a preempted phase. The preceding phase P_{n-1} is not a continued phase while the following phase P_{n+1} is also a preempted phase. Since P_n is not preceded by a continued phase all jobs of I_n are started in $S(P_n)$ and $t_S(i) = t_{S'}(i)$, for all $i \in I_n$. We obtain

$$ALG2_k(\sigma) = \sum_{i \in I_n} w(t_{S'}(i), \delta_i, v_i).$$

Let $i_p \in I_n$ be the job preempted in $S(P_n)$. The job is preempted at the end of $S(P_n)$. Moreover, its per-unit value is strictly smaller than the per-unit value of any job scheduled in $S(P_{n+1})$, the schedule of the following phase, since otherwise $ALG2_k$ would have scheduled job i_p in $S(P_{n+1})$. Phase P_{n+1} is also a preempted phase and the job preempted in $S(P_{n+1})$ is scheduled at the end of $S(P_{n+1})$. Thus the job preempted in $S(P_{n+1})$ has a strictly smaller per-unit value than any job scheduled in $S(P_{n+2})$. It follows that job i_p has a strictly smaller per-unit value than any job scheduled in $S(P_{n+1})$ and $S(P_{n+2})$. Lemma 7 ensures that *CHOP* does not schedule i_p in phases P_{n+1} and P_{n+2}. Thus the value achieved by *CHOP* for the preempted portion of i_p is upper bounded by $w((n+2)k, \infty, v_{i_p})$ and

$$\begin{aligned} CHOP(\sigma) &\leq \sum_{i \in I_n} w(t_{S'}(i), \delta_i, v_i) + w((n+2)k, \infty, v_{i_p}) \\ &= ALG2_k(\sigma) + w((n+2)k, \infty, v_{i_p}). \end{aligned}$$

Recall that job i_p is one having the smallest per-unit value among jobs scheduled in $S(P_n)$. Thus $ALG2_k(\sigma) \geq w((n-1)k, k, v_{i_p}) = \beta^{(n-1)k}(1-\beta^k)/(1-\beta) \cdot v_{i_p}$. Furthermore $w((n+2)k, \infty, v_{i_p}) = \beta^{(n+2)k}/(1-\beta)v_{i_p}$. We conclude $CHOP(\sigma)/ALG2_k(\sigma) \leq 1 + \beta^{3k}/(1-\beta^k)$. □

Lemma 9 *Let σ be a segment consisting of at least two phases. If σ consists of two phases, assume that the first one is a preempted phase. If σ consists of three phases, assume that the per-unit value of the job preempted in the first phase is at least as high as that of the job preempted in the third phase. There holds $CHOP(\sigma)/ALG2_k(\sigma) \leq \max\{1/\beta^{k-1}, 1/(1-\beta^{2k})\}$.*

Proof We first study the case that σ consists of exactly two phases P_n and P_{n+1}. Since P_n is a preempted phase, by construction of 2-phase segments, P_n is not preceded by a continued phase. Since neither P_n nor P_{n+1} are preceded by a continued phase, all job of I_n and I_{n+1} are started in $S(P_n)$ and $S(P_{n+1})$, respectively, and the sets I'_n and I'_{n+1} are empty. Thus $t_S(i) = t_{S'}(i)$, for all $i \in I_n \cup I_{n+1}$. We obtain

$$ALG2_k(\sigma) = \sum_{i \in I_n} w(t_{S'}(i), \delta_{i,n}, v_i) + \sum_{i \in I_{n+1}} w(t_{S'}(i), \delta_{i,n+1}, v_i). \tag{1}$$

Since $t_S(i) = t_{S'}(i) \leq t_{S^\star}(i)$, for all $i \in I_n \cup I_{n+1}$, *CHOP* cannot achieve a higher value for the jobs and job portions scheduled by $ALG2_k$ in σ. It can only attain an additional value for the job portion that is preempted in $S(P_n)$. Let i_p be the preempted

job. This job is scheduled last in $S(P_n)$. It is one having the smallest per-unit value among the jobs scheduled in $S(P_n)$. Moreover, it has a strictly smaller per-unit value than any job scheduled in $S(P_{n+1})$ because otherwise $ALG2_k$ would have included job i_p in the schedule $S(P_{n+1})$. By Lemma 7 *CHOP* does not schedule job i_p in phase P_{n+1}. Since $t_{S'}(i_p) \leq t_{S^\star}(i_p)$, *CHOP* can schedule at most $\delta_{i_p,n}$ time units of job i_p before the end of P_n. Hence *CHOP*'s additional value obtained in processing the preempted portion of i_p is at most $w((n+1)k, \infty, v_{i_p})$. Thus

$$CHOP(\sigma) \leq ALG2_k(\sigma) + w((n+1)k, \infty, v_{i_p}). \tag{2}$$

In $S(P_n)$ and $S(P_{n+1})$ all jobs have a per-unit value of at least v_{i_p}. Hence $ALG2_k(\sigma) \geq \beta^{(n-1)k}(1-\beta^{2k})/(1-\beta) \cdot v_{i_p}$. Taking into account that $w((n+1)k, \infty, v_{i_p}) = \beta^{(n+1)k}/(1-\beta)v_{i_p}$ we conclude that $CHOP(\sigma)/ALG2_k(\sigma) \leq 1 + \beta^{2k}/(1-\beta^{2k}) = 1/(1-\beta^{2k})$.

We next consider the case that σ consists of three phases P_n, P_{n+1}, P_{n+2} and that the job preempted in the first phase has a per-unit value that is at least as high as that of the job preempted in the third phase. By the definition of a 3-phase segment, P_n is a preempted phase that is not preceded by a continued phase and P_{n+1} is a continued phase. This combination of phases was analyzed above. Let σ_1 be the subsegment of phases P_n and P_{n+1}. Using (1) and (2), we obtain

$$ALG2_k(\sigma_1) = \sum_{i \in I_n} w(t_{S'}(i), \delta_{i,n} v_i) + \sum_{i \in I_{n+1}} w(t_{S'}(i), \delta_{i,n+1} v_i)$$

and

$$CHOP(\sigma_1) \leq ALG2_k(\sigma_1) + w((n+1)k, \infty, v_{i_p}), \tag{3}$$

where i_p is again the job preempted in $S(P_n)$.

Let σ_2 be the subsegment consisting of the third phase P_{n+2} and let $i_1 = i^1_{n+2}$ be the job scheduled at the beginning of P_{n+2}. Then

$$\begin{aligned} ALG2_k(\sigma_2) &= w((n+1)k, \delta_{i_1,n+2}, v_{i_1}) + \sum_{i \in I'_{n+2}} w(t_{S'}(i) + \delta_{i_1,n+2}, \delta_i, v_i) \\ &\quad + \sum_{i \in I_{n+2} \setminus (I'_{n+2} \cup \{i_1\})} w(t_{S'}(i), \delta_i, v_i) \\ &\geq \beta^{k-1} \left(w((n+1)k, \delta_{i_1,n+2}, v_{i_1}) + \sum_{i \in I_{n+2} \setminus \{i_1\}} w(t_{S'}(i), \delta_i, v_i) \right). \end{aligned}$$

The inequality holds because any $i \in I'_{n+2}$ is delayed by at most $k-1$ time units and hence $w(t_{S'}(i) + \delta_{i_1,n+2}, \delta_i, v_i) = \beta^{\delta_{i_1,n+2}} w(t_{S'}(i), \delta_{i,n+2}, v_i) \geq \beta^{k-1} w(t_{S'}(i), \delta_{i,n+2}, v_i)$, for any $i \in I'_{n+2}$.

Let i'_p be the job preempted in $S(P_{n+2})$. By assumption this job has a per-unit value that is upper bounded by the per-unit value of i_p. The additional value achieved by

CHOP in scheduling the preempted portions of both i_p and i'_p is upper bounded by the term $w((n+1)k, \infty, v_{i_p})$ in (3), which is again the value of scheduling an infinitely long job of per-unit value v_{i_p} starting at time $(n+1)k$. By slightly abusing notation let

$$CHOP(\sigma_2) = w((n+1)k, \delta_{i_1,n}, v_{i_1}) + \sum_{i \in I_{n+2} \setminus \{i_1\}} w(t_{S'}(i), \delta_{i,n}, v_i)$$

be the value achievable by *CHOP* in processing the jobs and job portions of $S(P_{n+2})$. There holds $ALG2_k(\sigma) = ALG2_k(\sigma_1) + ALG2_k(\sigma_2)$ and $CHOP(\sigma) \leq CHOP(\sigma_1) + CHOP(\sigma_2)$. Hence $CHOP(\sigma)/ALG2_k(\sigma) \leq \max\{CHOP(\sigma_1)/ALG2_k(\sigma_1), CHOP(\sigma_2)/ALG2_k(\sigma_2), \}$. As shown above the first ratio is upper bounded by $1/(1-\beta^{2k})$. The second ratio is at most $1/\beta^{k-1}$. □

Lemma 10 *Let σ be a segment consisting of at least two phases. If σ consists of two phases, assume that the first one is a continued phase. If σ consists of three phases, assume that the per-unit value of the job preempted in the first phase is smaller than that of the job preempted in the third phase. There holds $CHOP(\sigma)/ALG2_k(\sigma) \leq \max\{1/\beta^{k-1}, 1/(1-\beta^{2k})\}$.*

Proof The techniques used to prove Lemma 10 are similar to those of the proof of Lemma 9. Here the analysis becomes slightly more involved because in case of a 2-phase segment both phases P_n and P_{n+1} may initially process jobs i_n^1 and i_{n+1}^1 that were already scheduled in the preceding phases. We will show that a worst case occurs if the per-unit values of i_n^1 and i_{n+1}^1 are roughly equal to those in I'_n and I'_{n+1}, respectively.

Formally, we first investigate a segment σ consisting of two phases P_n and P_{n+1}. By assumption, the first one is a continued phase. Let $i_1 = i_n^1$ and $i_2 = i_{n+1}^1$ be the jobs scheduled first in $S(P_n)$ and $S(P_{n+1})$, respectively. There holds

$$\begin{aligned} ALG2_k(\sigma) = {} & w((n-1)k, \delta_{i_1,n}, v_{i_1}) + \sum_{i \in I'_n} w(t_{S'}(i) + \delta_{i_1,n}, \delta_{i,n}, v_i) && (4) \\ & + \sum_{i \in I_n \setminus (I'_n \cup \{i_1\})} w(t_{S'}(i), \delta_{i,n}, v_i) \\ & + w(nk, \delta_{i_2,n+1}, v_{i_2}) + \sum_{i \in I'_{n+1}} w(t_{S'}(i) + \delta_{i_2,n+1}, \delta_{i,n+1}, v_i) && (5) \\ & + \sum_{i \in I_{n+1} \setminus (I'_{n+1} \cup \{i_2\})} w(t_{S'}(i), \delta_{i,n+1}, v_i). \end{aligned}$$

We next analyze the value that *CHOP* can achieve for the jobs and job portions scheduled in $S(P_n)$ and $S(P_{n+1})$. If in S the job i_1 is started before the beginning of P_n, then i_1 is started at the end of an earlier phase $P_{n'}$, with $n' < n$, and when $ALG2_k$ constructed $S(P_{n'})$ it did not delay the starting time of job i_1 in Step (2). Since $t_{S'}(i_1) \leq t_{S^\star}(i_1)$, *CHOP* cannot schedule the $\delta_{i_1,n}$ units of job i_1 processed by $ALG2_k$ in $S(P_n)$ before the start of P_n. For any other job $i \in I_n$, $i \neq i_1$, there holds

$t_{S'}(i) \leq t_{S^\star}(i)$. Thus *CHOP* cannot not start the jobs and job portions scheduled in $S(P_n)$ before the beginning of P_n. The highest value is achieved by sequencing the jobs and job portions in non-decreasing order of per-unit value. This is in fact the sequence used in $S'(P_n)$, which first processes the jobs of I'_n, then the $\delta_{i_1,n}$ time units of job i_1 and finally the jobs of $I_n \setminus I'_n$. Let $\Delta_1 = \sum_{i \in I'_n} \delta_{i,n}$. The value achieved by *CHOP* for the jobs and job portions scheduled in $S(P_n)$ is at most

$$\sum_{i \in I'_n} w(t_{S'}(i), \delta_{i,n}, v_i) + w((n-1)k + \Delta_1, \delta_{i_1,n}, v_{i_1}) + \sum_{i \in I_n \setminus (I'_n \cup \{i_1\})} w(t_{S'}(i), \delta_{i,n}, v_i).$$

An analogous expression holds for phase P_{n+1}. Let i_p be the job preempted in $S(P_{n+1})$. This job is scheduled last in $S'(P_{n+1})$ and since $t_{S'}(i_p) \leq t_{S^\star}(i_p)$ *CHOP* can finish at most $\delta_{i_p,n+1}$ units of this job before the end of P_{n+1}. Thus the additional value obtained by *CHOP* in processing the preempted portion of i_p is at most $w((n+1)k, \infty, v_{i_p})$. We conclude

$$\begin{aligned} CHOP(\sigma) \leq & \sum_{i \in I'_n} w(t_{S'}(i), \delta_{i,n}, v_i) + w((n-1)k + \Delta_1, \delta_{i_1,n}, v_{i_1}) && (6) \\ & + \sum_{i \in I_n \setminus (I'_n \cup \{i_1\})} w(t_{S'}(i), \delta_{i,n}, v_i) \\ & + \sum_{i \in I'_{n+1}} w(t_{S'}(i), \delta_{i,n+1}, v_i) + w(nk + \Delta_2, \delta_{i_2,n+1}, v_{i_2}) && (7) \\ & + \sum_{i \in I_{n+1} \setminus (I'_{n+1} \cup \{i_2\})} w(t_{S'}(i), \delta_{i,n+1}, v_i) + w((n+1)k, \infty, v_{i_p}). && (8) \end{aligned}$$

Here Δ_2 is the total length of the jobs in I'_{n+1}. Define $R = CHOP(\sigma)/ALG2_k(\sigma)$. In order to upper bound R we consider the ratios restricted to the job sets I'_n and I'_{n+1}. If $I'_n \neq \emptyset$, let

$$r_1 = \sum_{i \in I'_n} w(t_{S'}(i), \delta_{i,n}, v_i) \Big/ \sum_{i \in I'_n} w(t_{S'}(i) + \delta_{i_1,n}, \delta_{i,n}, v_i).$$

If $I'_{n+1} \neq \emptyset$, let

$$r_2 = \sum_{i \in I'_{n+1}} w(t_{S'}(i), \delta_{i,n+1}, v_i) \Big/ \sum_{i \in I'_{n+1}} w(t_{S'}(i) + \delta_{i_2,n+1}, \delta_{i,n+1}, v_i).$$

If $I'_n \neq \emptyset$, then $r_1 = 1/\beta^{\delta_{i_1,n}} \leq 1/\beta^{k-1}$. Similarly, if $I'_{n+1} \neq \emptyset$, then $r_2 = 1/\beta^{\delta_{i_2,n+1}} \leq 1/\beta^{k-1}$.

If R is not upper bounded by $1/\beta^{k-1}$, then R is increasing for decreasing values of v_i, with $i \in I'_n \cup I'_{n+1}$. Recall that each v_i, with $i \in I'_n$, is greater than v_{i_1}; each v_i, with $i \in I'_{n+1}$, is greater than v_{i_2}. Hence R is either upper bounded by $1/\beta^{k-1}$ or by the ratio

obtained when setting $v_i = v_{i_1}$, for $i \in I'_n$, and $v_i = v_{i_2}$, for $i \in I'_{n+1}$. With this setting $\sum_{i \in I'_n} w(t_{S'}(i) + \delta_{i_1,n}, \delta_{i,n}, v_i) = w((n-1)k + \delta_{i_1,n}, \Delta_1, v_{i_1})$ and the right-hand side expression of (4) becomes $w((n-1)k, \delta_{i_1,n} + \Delta_1, v_{i_1})$. Similarly, the expression (5) becomes $w(nk, \delta_{i_2,n+1} + \Delta_2, v_{i_2})$. Analogously, $\sum_{i \in I'_n} w(t_{S'}(i), \delta_{i,n}, v_i) = w((n-1)k, \Delta_1, v_{i_1})$ and the right-hand side of expression (6) is $w((n-1)k, \delta_{i_1,n} + \Delta_1, v_{i_1})$. Expression (7) becomes $w(nk, \delta_{i_2,n+1} + \Delta_2, v_{i_2})$. With these observations, $ALG2_k(\sigma)$ simplifies to

$$V = w((n-1)k, \delta_{i_1,n} + \Delta_1, v_{i_1}) + \sum_{i \in I_n \setminus (I'_n \cup \{i_1\})} w(t_{S'}(i), \delta_{i,n}, v_i)$$
$$+ w(nk, \delta_{i_2,n+1} + \Delta_2, v_{i_2}) + \sum_{i \in I_{n+1} \setminus (I'_{n+1} \cup \{i_2\})} w(t_{S'}(i), \delta_{i,n+1}, v_i)$$

and $CHOP(\sigma)$ simplifies to $V + w((n+1)k, \infty, v_{i_p})$. If R is not upper bounded by $1/\beta^{k-1}$, then it is upper bounded by $R' = (V + w((n+1)k, \infty, v_{i_p}))/V$. Recall that P_n is a continued phase. The job scheduled last in $S(P_n)$ is one having the smallest per-unit value among jobs scheduled in $S(P_n)$. Since the job is also scheduled in $S(P_{n+1})$, it has a per-unit value that is at least as high as that of job i_p. This preempted job is among the jobs having the smallest per-unit value in $S(P_{n+1})$. We conclude that all jobs scheduled in $S(P_n)$ and $S(P_{n+1})$ have a per-unit value of at least v_{i_p}. This implies $V \geq \beta^{(n-1)k}(1-\beta^{2k})/(1-\beta)v_{i_p}$. Furthermore $w((n+1)k, \infty, v_{i_p}) = \beta^{(n+1)k}/(1-\beta)v_{i_p}$. We conclude that $R' \leq 1 + \beta^{2k}/(1-\beta^{2k}) = 1/(1-\beta^{2k})$.

Finally suppose that σ consists of three phases. Since a 3-phase segments ends with a continued phase followed by a preempted case we can reduce the analysis to that developed above. Let P_{n-1} be the first phase of σ. Since P_{n-1} is not preceded by a continued phase, all jobs scheduled in $S(P_{n-1})$ are also started in this phase. Moreover, when $ALG2_k$ constructs $S(P_{n-1})$ no job is delayed in Step (2) in order to move some job to the beginning of the phase. As $t_S(i) = t_{S'}(i) \leq t_{S^\star}(i)$, for all $i \in I_{n-1}$, the value achieved by *CHOP* in scheduling the jobs and job portions of $S(P_{n-1})$ is not higher than the value attained by $ALG2_k$. Let job i'_p be the one preempted in $S(P_{n-1})$. Since the job is not scheduled in $S(P_n)$ it has a strictly smaller per-unit value than any job processed in $S(P_n)$. By assumption it has a smaller per-unit value than job i_p. Hence the per-unit value of job i'_p is strictly smaller than that of any job sequenced in $S(P_n)$ and $S(P_{n+1})$. Lemma 7 implies that *CHOP* does not schedule job i'_p in P_n and P_{n+1}. Thus *CHOP* schedules the preempted portion of job i'_p no earlier than time $(n+1)k$. It follows that the total additional value attained by *CHOP* for the preempted portions of jobs i'_p and i_p is at most $w((n+1)k, \infty, v_{i_p})$, which is the last term in (8). Since for the jobs and job portions scheduled in $S(P_{n-1})$ *CHOP* does not achieve a higher value than $ALG2_k$, it follows that $CHOP(\sigma)/ALG2_k(\sigma)$ is upper bounded by the ratio R analyzed above. □

We proceed with the proof of Theorem 3.

Proof of Theorem 3 Lemma 8 addresses segments consisting of a single phase. Lemmas 9 and 10 cover all possible cases of 2-phase and 3-phase segments.

Together the three lemmas imply $CHOP(\sigma)/ALG2_k(\sigma) \leq \max\{1/\beta^{k-1}, 1/(1-\beta^{2k}), 1+\beta^{3k^*}/(1-\beta^{k^*})\}$, for any segment σ. Hence $CHOP(\mathcal{I}_k)/ALG2_k(\mathcal{I}_k)$ and $OPT(\mathcal{I}_k)/ALG2_k(\mathcal{I}_k)$ are each upper bounded by $\max\{1/\beta^{k-1}, 1/(1-\beta^{2k}), 1+\beta^{3k^*}/(1-\beta^{k^*})\}$. The theorem follows because $1/\beta^{k-1}OPT(\mathcal{I}_k) \geq OPT(\mathcal{I})$ and $ALG2_k(\mathcal{I}_k) = ALG2_k(\mathcal{I})$. □

We conclude our analysis by determining the best value for the phase length k.

Corollary 4 *Let $c = 1+\phi$, where $\phi = (1+\sqrt{5})/2$ is the Golden Ratio. For $k = \lfloor -\frac{1}{2}\log_\beta c \rfloor + 1$, $ALG2_k$ achieves a competitive ratio of $c \approx 2.618$.*

Proof Let $k^* = \lfloor -\frac{1}{2}\log_\beta c \rfloor + 1$. We evaluate the competitive ratio defined in Theorem 3. First, $1/\beta^{k^*-1} \leq \sqrt{c}$ and hence $1/\beta^{2(k^*-1)} \leq c$. We show that $1/(1-\beta^{2k^*})$ and $\beta^{3k^*}/(1-\beta^{k^*})$ are each upper bounded by $\sqrt{c}$. There holds $1/(1-\beta^{2k^*}) \leq c/(c-1) = c/\sqrt{c} = \sqrt{c}$. Furthermore, $\beta^{3k^*}/(1-\beta^{k^*}) \leq 1/(c\sqrt{c}-c)$. In order to show $1+\beta^{3k^*}/(1-\beta^{k^*}) \leq \sqrt{c}$ it suffices to verify that $1 \leq c - \sqrt{c}$. The last inequality is satisfied because $c - \sqrt{c} = 1$. □

4 An Algorithm for Multiple Ad Positions

We study the setting where m parallel machines (ad positions) are available. Let $\mathcal{I} = (a_i, v_i, l_i)_{i=1}^N$ be any input. Each job may be processed on one machine only, i.e. a migration of jobs is not allowed. We define an algorithm $ALG(m)_k$ that generalizes $ALG1_k$ of Sect. 2. Let $P_n = (n-1)k, \ldots, nk-1$ be any phase. Again let Q_n be the set of jobs i that have arrived until the beginning of P_n, i.e. $a_i \leq (n-1)k$, and have not been processed in the past phases $P_1, \ldots, P_{n-1}$. $ALG(m)_k$ constructs a schedule for P_n by considering the k time steps of the phase. For $t = (n-1)k, \ldots, nk-1$, $ALG(m)_k$ determines the m jobs having the highest per-unit values among the jobs of Q_n that are unfinished at time t. Each of these jobs is scheduled for one time unit. If a job was also scheduled at time $t-1$, then it is assigned to the same machine at time t. We specify a tie breaking rule if, among the unfinished jobs in Q_n, the m-th largest per-unit value is v and there exist several jobs having this value. In this case preference is given to those jobs that have already been started at times t', with $t' < t$. Jobs that have not been started yet are considered in increasing order of arrival time, where ties may be broken arbitrarily. Of course, if at time t set Q_n contains at most m unfinished jobs, then each of them is scheduled at that time. We observe that a feasible phase schedule, in which each job is processed without interruption on one machine, can be constructed easily: If a job of Q_n is among those having the m highest per-unit values, then the job will remain in this subset until it is finished. Hence the job can be sequenced continuously on the same machine. We also observe that on each machine jobs are sequenced in order of non-increasing per-unit value. Let $S(P_n)$ denote the schedule constructed for phase P_n. While $S(P_n)$ is executed, newly arriving jobs are deferred until the beginning of the next phase. At the end of $S(P_n)$ unfinished jobs are preempted. A pseudo-code description of the algorithm is given in Fig. 3.

Theorem 4 *For all $k \in \mathbb{N}$ and all probabilities β, $ALG(m)_k$ achieves a competitive ratio of $\frac{1}{\beta^{k-1}}(1+\frac{1}{1-\beta^k})$.*

Algorithm $ALG(m)_k$: Each phase P_n is processed as follows.
At any time t, $(n-1)k \leq k \leq nk-1$, schedule the m unfinished jobs with the highest per-unit value in Q_n for one time unit each. At the end of P_n preempt the unfinished jobs. Execute this schedule $S(P_n)$, ignoring jobs that arrive during the phase.

Fig. 3 The algorithm $ALG(m)_k$

In the following we prove Theorem 4. Unfortunately, we cannot extend the analysis of $ALG1_k$ given in Sect. 2 because Lemma 2 does not hold for schedules produced by m-machine generalizations of *CHOP*. Therefore we have to resort to different "optimal" schedules.

As always, given any input $\mathcal{I} = (a_i, v_i, l_i)_{i=1}^N$, we consider the k-quantized input $\mathcal{I}_k = (a'_i, v_i, l_i)_{i=1}^N$ with $a'_i = k\lceil a_i/k \rceil$. There holds $ALG(m)_k(\mathcal{I}) = ALG(m)_k(\mathcal{I}_k)$ and $1/\beta^{k-1} OPT(\mathcal{I}_k) \geq OPT(\mathcal{I})$, which can be shown in the same way as in Lemma 1. Let S denote the schedule constructed by $ALG(m)_k$ for $\mathcal{I}_k$. Furthermore, let S_{OPT} be an optimal schedule for $\mathcal{I}_k$.

The main idea of the proof of Theorem 4 is to construct a schedule $S^\star$ whose value is at least as high as that of S_{OPT} and that will allow us to compare the values of S and $S^\star$. Schedule $S^\star$ can be derived from S_{OPT} and is given extra ability to process $\mathcal{I}_k$. In $S^\star$ jobs may be interrupted and migrated among machines. Moreover $S^\star$ may process up to $2m$ jobs at any time t.

For any time t, let I_t^* denote the set of jobs scheduled in $S^\star$ at time t. Similarly, let I_t be the set of jobs processed in S at time t. Set $I_t^* \setminus I_t$ contains those jobs that are only scheduled in $S^\star$ but not in S at that time. We will define a matching that matches some jobs of I_t^* with those in I_t. Using this matching we can upper bound the extra value achieved by $S^\star$ relative to the value of S. Schedule $S^\star$ and the associated matching satisfy a specific property. For any time t, each job $i \in I_t^* \setminus I_t$ is either matched with a job $i' \in I_t$ whose per-unit value is at least as high as that of job i, or job i is scheduled in S at the end of an earlier phase P_n, i.e., at some time $t' = nk-1$ where $nk-1 < t$. Each $i' \in I_t$ is matched with at most one $i \in I_t^*$. We remark that the matching is time dependent. Over time a job scheduled in S or $S^\star$ may be matched with various jobs. However, for any fixed time t, each $i' \in I_t$ is matched with at most one $i \in I_t^*$ and vice versa. Formally, schedules $S^\star$ and S, together with the sets I_t^* and I_t, satisfy the following property (P).

(P) For all times t, each job $i \in I_t^* \setminus I_t$ is either matched with an $i' \in I_t$ such that $v_{i'} \geq v_i$ or it is scheduled in S at the end of a previous phase P_n, i.e., at some time $nk-1$ where $nk-1 < t$. For all times t, each $i' \in I_t$ is matched with at most one $i \in I_t^*$.

Let $OPT^*(\mathcal{I}_k)$ denote the value of schedule $S^\star$.

Lemma 11 *There exists a schedule $S^\star$ satisfying property (P) and $OPT^*(\mathcal{I}_k) \geq OPT(\mathcal{I}_k)$.*

Proof We transform S_{OPT} into a schedule satisfying the statement of the lemma. First consider the schedule S. Recall that $ALG(m)_k$ schedules each job of $\mathcal{I}_k$ in only one phase, i.e., no job is processed in two or more phases. For any phase P_n, $n \geq 1$, let

I_n be the set of jobs that are sequenced in schedule S in P_n but not at the end of this phase at time $nk-1$. We observe that each job $i \in I_n$ is scheduled completely, i.e. for l_i time units, in P_n: Algorithm $ALG(m)_k$ always schedules the jobs having the highest per-unit values among the available unfinished jobs. If a job is scheduled at time t but not at time $t+1$ and t is not the end of a phase, then the job is completely processed because in $\mathcal{I}_k$ no jobs arrive during a phase. Let $t_S(i)$ be the starting time of job i in S. Then job $i \in I_n$ is scheduled at times $t_S(i), \ldots, t_S(i)+l_i-1$ is S. Let $I = \cup_{n\geq 1} I_n$.

We now modify S_{OPT} as follows. For each job $i \in I$ and each time t with $t_S(i) \leq i \leq t_S(i)+l_i-1$, one unit of job i is scheduled at time t provided that S_{OPT} did not already schedule this job time t. Furthermore, units of job i sequenced in S_{OPT} at times $t \geq t_S(i)+l_i$ are deleted. These modifications do not decrease the value of the schedule because each job unit deleted at time t is replaced by a corresponding unit scheduled at some time $t' < t$. Let $S^\star$ be the resulting schedule. Note that in $S^\star$ up to $2m$ jobs can be scheduled at any time.

For any time t, let $\Delta_t = I_t^* \setminus I_t$ be the set of jobs that are only scheduled in $S^\star$ but not in S at time t. We partition this set into two subsets. Let $\Delta_t^1 \subseteq \Delta_t$ be the set of jobs that are scheduled at times $t' < t$ in S. Set $\Delta_t^2 = \Delta_t \setminus \Delta_t^1$ contains the remaining jobs. We first study Δ_t^1 and argue that each $i \in \Delta_t^1$ must be scheduled in S at the end of a phase P_n with $nk-1 < t$. By the definition of Δ_t^1, job i is not scheduled in S at time t but is scheduled in S at some time $t' < t$. Let P_n be the phase containing time t'. If job i is not scheduled in S at the end of P_n, then $i \in I_n$. However the schedule modifications described in the last paragraph deleted units of job i sequenced after $t_S(i)+l_i-1$ and hence at time t. We conclude that each $i \in \Delta_t^1$ satisfies property (P).

It remains to consider the jobs of Δ_t^2. None of these jobs is scheduled in S at times t' with $t' \leq t$. Since the jobs of Δ_t^2 are available for processing at time t, schedule S sequences exactly m jobs at time t. Recall again that $ALG(m)_k$ always processes the available unfinished jobs with the highest per-unit values. Thus I_t contains $|\Delta_t^2|$ jobs whose per-unit values are at least as high as the per-unit value of each job in Δ_t^2. These jobs of I_t may be matched in an arbitrary way with the jobs of Δ_t^2. This establishes property (P) for any $i \in \Delta_t^2$ and thus for all jobs of Δ_t. □

Lemma 12 *For all $k \in \mathbb{N}$ and all probabilities β, inequality $\left(1+\frac{1}{1-\beta^k}\right) ALG(m)_k(\mathcal{I}_k) \geq OPT^*(\mathcal{I}_k)$ holds.*

Proof For any phase P_n, $n \geq 1$, let $ALG(m)_k(P_n)$ denote the value obtained in S during phase P_n. Similarly, $OPT^*(P_n)$ is the value achieved in $S^\star$ during P_n. There holds $ALG(m)_k(\mathcal{I}_k) = \sum_{n\geq 1} ALG(m)_k(P_n)$ and $OPT^*(\mathcal{I}_k) = \sum_{n\geq 1} OPT^*(P_n)$.

In $S^\star$ consider any time t. Let $I_{t,1}^* \subseteq I_t^*$, be the set of jobs that are also scheduled in S at time t or that are matched to a job in I_t. Set $I_{t,2}^* = I_t^* \setminus I_{t,1}^*$ consists of the remaining jobs of I_t^*. By the definition of $I_{t,1}^*$ and property (P), each job of $I_{t,2}^*$ is scheduled in S at the end of a previous phase, i.e. at some time $t' = nk-1$ with $t' < t$. Let $OPT_1^*(t) = \sum_{i\in I_{t,1}^*} \beta^t v_i$ and $OPT_2^*(t) = \sum_{i\in I_{t,2}^*} \beta^t v_i$ be the values obtained in scheduling jobs of $I_{t,1}^*$ and $I_{t,2}^*$ at time t. By summing the latter quantities over all times t of a phase P_n, we let $OPT_1^*(P_n) = \sum_{t=(n-1)k}^{nk-1} OPT_1^*(t)$ and $OPT_2^*(P_n) = \sum_{t=(n-1)k}^{nk-1} OPT_2^*(t)$. There holds $OPT^*(P_n) = OPT_1^*(P_n) + OPT_2^*(P_n)$.

Let $ALG(m)_k(t) = \sum_{i \in I_t} \beta^t v_i$ be the value obtained in schedule S a time t. Again $ALG(m)_k(P_n) = \sum_{t=(n-1)k}^{nk-1} ALG(m)_k(t)$ is the total value for phase P_n. By the definition of $I^*_{t,1}$, each job of this set is contained in I_t or is matched with a $i' \in I_t$ satisfying $v_i \leq v_{i'}$. Hence $OPT^*_1(t) \leq 2ALG(m)_k(t)$ and, by summing is inequality over all times of phase P_n, we obtain $OPT^*_1(P_n) \leq 2ALG(m)_k(P_n)$. It follows

$$\begin{aligned} OPT^*(\mathcal{I}_k) &= \sum_{n\geq 1} OPT^*_1(P_n) + \sum_{n\geq 1} OPT^*_2(P_n) \;\leq\; \sum_{n\geq 1} 2ALG(m)_k(P_n) + \sum_{n\geq 1} OPT^*_2(P_n) \\ &= 2ALG(m)_k(\mathcal{I}_k) + \sum_{n\geq 1} OPT^*_2(P_n). \end{aligned}$$

We next upper bound $\sum_{n\geq 1} OPT^*_2(P_n)$. For any $n \geq 1$, let L_n be the set of jobs scheduled in S at the end of P_n, i.e., at time $kn - 1$. For any time t, each job $i \in I^*_{t,2}$ is contained in some set L_n with $nk - 1 < t$. Thus $\sum_{n\geq 1} OPT^*_2(P_n)$ is upper bounded by the total value obtained in scheduling jobs of L_n in $S^\star$ at times $t > nk - 1$, for all $n \geq 1$. The maximum possible value achievable in scheduling any job $i \in L_n$ at times $t > nk - 1$ is at most $\beta^{nk}/(1-\beta) \cdot v_i$, which is obtained if the job is scheduled for an infinite duration starting at time nk. We conclude

$$OPT^*(\mathcal{I}_k) \leq 2ALG(m)_k(\mathcal{I}_k) + \sum_{n\geq 1} \frac{\beta^{nk}}{1-\beta} \sum_{i\in L_n} v_i$$

and

$$\frac{OPT^*(\mathcal{I}_k)}{ALG(m)_k(\mathcal{I}_k)} \leq 2 + \left(\sum_{n\geq 1} \frac{\beta^{nk}}{1-\beta} \sum_{i\in L_n} v_i\right) \Bigg/ \left(\sum_{n\geq 1} ALG(m)_k(P_n)\right).$$

We finally show that, for any $n \geq 1$, there holds $(\frac{\beta^{nk}}{1-\beta} \sum_{i\in L_n} v_i)/ALG(m)_k(P_n) \leq \beta^k/(1-\beta^k)$. This implies $OPT^*(\mathcal{I}_k)/ALG(m)_k(\mathcal{I}_k) \leq 2 + \beta^k/(1-\beta^k) = 1 + 1/(1-\beta^k)$ and the lemma follows.

Consider again any phase P_n. Each $i \in L_n$ is scheduled in S at the end of P_n, i.e., at time $nk - 1$. When describing $ALG(m)_k$ we argued that in each phase and on each machine jobs are sequenced in non-increasing order of per-unit value. Hence on each machine where a job $i \in L_n$ is scheduled in S, jobs of per-unit value of at least v_i are sequenced throughout P_n. Since P_n consists of time steps $(n-1)k, \ldots, nk - 1$, we obtain $ALG(m)_k(P_n) \geq (\beta^{(n-1)k} - \beta^{nk})/(1-\beta) \cdot \sum_{i\in L_n} v_i$ and, as desired, $(\frac{\beta^{nk}}{1-\beta} \sum_{i\in L_n} v_i)/ALG(m)_k(P_n) \leq \beta^{nk}/(\beta^{(n-1)k} - \beta^{nk}) = \beta^k/(1-\beta^k)$. □

We are ready to prove Theorem 4.

Proof of Theorem 4 Lemmas 11 and 12 imply $(1 + 1/(1-\beta^k))ALG(m)_k(\mathcal{I}_k) \geq OPT^*(\mathcal{I}_k) \geq OPT(\mathcal{I}_k)$, for any input $\mathcal{I}$. Since $ALG(m)_k(\mathcal{I}) = ALG(m)_k(\mathcal{I}_k)$ and $1/\beta^{k-1} OPT(\mathcal{I}_k) \geq OPT(\mathcal{I})$, the theorem follows. □

We finally determine the best choice of k. For any β, the competitive ratio of Corollary 5 is upper bounded by $(1+1/(\sqrt{2}-1))/(2-\sqrt{2}) = 1/(3-2\sqrt{2}) \approx 5.828$.

Corollary 5 *For $k = \lceil \log_\beta(2-\sqrt{2}) \rceil$, the resulting algorithm $ALG(m)_k$ achieves a competitive ratio of $(1 + 1/(1 - \beta(2 - \sqrt{2}))/(2 - \sqrt{2})$.*

Proof The function $f(x) = \frac{1}{\beta^{x-1}}(1 + \frac{1}{1-\beta^x})$ is minimized for $x^* = \log_\beta(2 - \sqrt{2})$. Furthermore it is increasing for $x \geq x^*$. Thus, for $k = \lceil \log_\beta(2-\sqrt{2}) \rceil$, the competitive ratio of $ALG(m)_k$ is upper bounded by $f(x^* + 1) = (1 + 1/(1 - \beta(2 - \sqrt{2}))/(2 - \sqrt{2})$. □

5 Conclusions

In this paper we have presented new algorithms for storyboarding. As a main contribution we have developed a deterministic online algorithm that achieves a competitive factor of $1+\phi$, where $\phi = (1+\sqrt{5})/2$ is the Golden Ratio. The resulting performance guarantee of approximately 2.618 is close to the lower bound of 2 currently known for deterministic online strategies. Furthermore, we have studied for the first time a problem setting where stories may be shown simultaneously on several ad positions of a web page. We have devised an algorithm whose competitiveness is upper bounded by $1/(3 - 2\sqrt{2}) \approx 5.828$.

A natural direction for future work is to improve the above bounds. For the scenario where a single ad position is available, it would be exciting to develop randomized online algorithms whose competitiveness beats the deterministic lower bound. To this end one might investigate strategies that operate again in phases but determine the length of a phase according to a probability distribution. For the setting where multiple ad positions are available, we do not know any good lower bounds on the performance of online algorithms. It would be interesting to construct a lower bound that is higher than the best upper bound currently known for a single ad position. Another open problem is to extend the refined algorithm of Sect. 3 to this parallel setting. The technical issue is to construct consistent schedules in which jobs scheduled last in a phase are continued in the next phase. Furthermore, it would be interesting to explore randomized strategies.

Acknowledgements We thank an anonymous referee for helpful comments improving the presentation of the paper.

References

1. Morrell, L.: Global digital ad spend to hit $285bn by 2020. MarketingTechNews (June 2016). http://www.marketingtechnews.net/news/2016/jun/21/global-digital-ad-spend-hit-285-billion-2020/

2. Digital Ad spending to surpass TV next year. eMarketer article (March 2016). https://www.emarketer.com/Article/Digital-Ad-Spending-Surpass-TV-Next-Year/1013671
3. Dasgupta, A., Ghosh, A., Nazerzadeh, H., Raghavan, P.: Online story scheduling in web advertising. In: Proceedings of the 20th annual ACM-SIAM symposium on discrete algorithms, pp. 1275–1284 (2009)
4. Surround session described at marketingterms.com. http://www.marketingterms.com/dictionary/surround_session/
5. Sleator, D.D., Tarjan, R.E.: Amortized efficiency of list update and paging rules. Commun. ACM **28**, 202–208 (1985)
6. Balseiro, S.R., Feldman, J., Mirrokni, V.S., Muthukrishnan, S.: Yield optimization of display advertising with ad exchange. Manag. Sci. **60**(12), 2886–2907 (2014)
7. Buchbinder, N., Feldman, M., Ghosh, A., Naor, J.: Frequency capping in online advertising. J. Sched. **17**(4), 385–398 (2014)
8. Buchbinder, N., Jain, K., Naor, J.: Online primal-dual algorithms for maximizing ad-auctions revenue. In: Proceedings of the 15th annual European symposium on algorithms (ESA), Springer LNCS, vol. 4698, pp. 253–264 (2007)
9. Feldman, J., Korula, N., Mirrokni, V.S., Muthukrishnan, S., Pál, M.: Online ad assignment with free disposal. In: Proceedings of the 5th international workshop on internet and network economics (WINE), Springer LNCS, vol. 5929, pp. 374–385 (2009)
10. Feige, U., Immorlica, N., Mirrokni, V.S., Nazerzadeh, H.: A combinatorial allocation mechanism with penalties for banner advertising. In: Proceedings of the 17th international conference on world wide web, pp. 169–178 (2008)
11. Feldman, J., Mehta, A., Mirrokni, V.S., Muthukrishnan, S.: Online stochastic matching: beating 1-1/e. In: Proceedings of the 50th annual IEEE symposium on foundations of computer science, pp. 117–126 (2009)
12. Ghosh, A., Sayedi, A.: Expressive auctions for externalities in online advertising. In: Proceedings of the 19th international conference on world wide web, pp. 371-380 (2010)
13. Mehta, A.: Online matching and ad allocation. Found. Trends Theor. Comput. Sci. **8**(4), 265–368 (2013)
14. Mehta, A., Saberi, A., Vazirani, U.V., Vazirani, V.V.: AdWords and generalized online matching. J. ACM **54**(5), 22 (2007)