Online Disjoint Set Covers: Randomization is not Necessary

Marcin Bienkowski
University of Wrocław, Poland

Jarosław Byrka
University of Wrocław, Poland

Łukasz Jeż
University of Wrocław, Poland

Abstract

In the online disjoint set covers problem, the edges of a hypergraph are revealed online, and the goal is to partition them into a maximum number of disjoint set covers. That is, n nodes of a hypergraph are given at the beginning, and then a sequence of hyperedges (subsets of $[n]$) is presented to an algorithm. For each hyperedge, an online algorithm must assign a color (an integer). Once an input terminates, the gain of the algorithm is the number of colors that correspond to valid set covers (i.e., the union of hyperedges that have that color contains all n nodes).

We present a deterministic online algorithm that is $O(\log^2 n)$-competitive, exponentially improving on the previous bound of $O(n)$ and matching the performance of the best randomized algorithm by Emek et al. [ESA 2019].

For color selection, our algorithm uses a novel potential function, which can be seen as an online counterpart of the derandomization method of conditional probabilities and pessimistic estimators. There are only a few cases where derandomization has been successfully used in the field of online algorithms. In contrast to previous approaches, our result extends this tool to tackle the following new challenges: (i) the potential function derandomizes not only the Chernoff bound, but also the coupon collector's problem, (ii) the value of Opt of the maximization problem is not bounded a priori, and (iii) we do not produce a fractional solution first, but work directly on the input.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Disjoint Set Covers, Derandomization, pessimistic Estimator, potential Function, online Algorithms, competitive Analysis

Funding Supported by Polish National Science Centre grants 2022/45/B/ST6/00559, 2020/39/B/ST6/01641, and 2020/39/B/ST6/01679.

1 Introduction

In this paper, we study online algorithms for maximizing the number of set covers of a set of nodes. We focus on a hypergraph (set system) $G = (V, E)$ that has $n = |V|$ nodes and where each hyperedge $S \in E$ is a subset of nodes from V. A subset $E' \subseteq E$ is called *set cover* if $\bigcup_{S \in E'} S = V$, i.e., every node is covered by at least one hyperedge of E'. In the *disjoint set covers* (DSC) problem [13], the goal is to partition the set of hyperedges E into *maximum number* of mutually disjoint subsets $E = E_1 \uplus E_2 \uplus \cdots \uplus E_k$, where each E_j is a set cover. Note that E is a multi-set, i.e., it can contain multiple copies of the same hyperedge.

The problem has been studied in a theoretical setting, but as we discuss later, it also finds applications in energy minimization in sensor networks [13].

Coloring Perspective. When constructing a solution to the DSC problem, it is convenient to think in terms of coloring hyperedges.[1] Each color corresponds then to a subset of hyperedges that have that color, and the color is *fully used* if the hyperedges colored with it form a set cover. The problem then becomes equivalent to coloring hyperedges, so that the number of fully used colors is maximized.

Online Variant. In this paper, we focus on the *online* variant of the DSC problem, where the set V is given in advance, but the hyperedges of E arrive in an online fashion. Once a hyperedge $S \in E$ arrives, it must be colored immediately and irrevocably. Again, the goal is to maximize the number of fully used colors. The performance of an online algorithm is measured by the competitive ratio, i.e., the ratio of the number of fully used colors produced by the optimal *offline* algorithm OPT to that of an online algorithm ALG.

Our Contribution. We present a deterministic online $O(\log^2 n)$-competitive algorithm DET for the DSC problem, which improves exponentially over the $O(n)$-competitive algorithm by Emek et al. [15], and matches the performance of their randomized algorithm [15].

Our algorithm is based on a novel potential function that guides the choice of colors. As we discuss later in Subsection 1.3, our approach can be seen as an online-computable counterpart of the method of conditional probabilities that simultaneously controls the derandomization of the Chernoff bound and the coupon collector's problem.

1.1 Offline Scenario: Previous Results

The disjoint sets cover problem is known to be NP-complete [13], can be approximated within a factor of $(1 + o(1)) \cdot \ln |V|$ [21] and cannot be approximated within a factor of $(1 - \varepsilon) \cdot \ln |V|$ for any $\varepsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\log \log n})$ [16].[2]

OPT vs. Min-degree. We use $\text{OPT}(E)$ to denote the maximum number of disjoint set covers of a hypergraph $G = (V, E)$. This value is also called *cover-decomposition number* [6]. We denote the minimum degree of the hypergraph $G = (V, E)$ as $\delta(E) \triangleq \min_{i \in V} |S \in E : i \in S|$. Note that trivially $\text{OPT}(E) \geqslant \delta(E)$. While this bound may not be tight (cf. Subsection 1.4), $\delta(E)$ serves as a natural benchmark for approximation and online algorithms.

Random Coloring and its Straightforward Derandomization. An $O(\log n)$-approximation algorithm for the offline DSC problem [22] can be obtained by coloring each hyperedge with a color chosen uniformly at random from the palette of $\Theta(\delta(E)/\log n)$ colors. To analyze this approach, focus on a single node $i \in V$. We say that node i *gathers* color r if i is incident to a hyperedge colored with r. Since node i is contained in $\deg(v) \geqslant \delta(E)$ hyperedges, and there are $\Theta(\delta(E)/\log n)$ available colors, i gathers all palette colors with high probability. By the union bound, this property holds for all nodes, i.e., all $\Theta(\delta(E)/\log n)$ colors are fully used by an algorithm (also with high probability). Since $\text{OPT}(E) \leqslant \delta(E)$, the approximation ratio of $O(\log n)$ follows.

The hyperedges can be processed in a fixed order, and the random choice of a color can be replaced by the deterministic one by a simple use of the method of conditional probabilities [22, 3].

[1] The DSC problem should not be confused with the hyperedge coloring problem, which requires that the hyperedges of the same color are disjoint.
[2] The authors of [16] call this problem *set cover packing* or *one-sided domatic problem*.

	Upper bound	Lower bound
Randomized	$O(\log^2 n)$ [15]	$\Omega(\log n / \log\log n)$ [15]
Deterministic	$O(n)$ [15] $\mathbf{O(\log^2 n)}$ (Theorem 6)	$\Omega(\log n / \log\log n)$ [15]

Table 1 Previous and new bounds on the competitive ratios for the online DSC problem.

1.2 Online scenario: previous results

In the online case, the edges of E are revealed sequentially. For notational convenience, we use E to denote both the set and a sequence of hyperedges (the input). We use $\textsc{Alg}(E)$ to denote the number of fully used colors in the solution of an online algorithm $\textsc{Alg}$.

Competitive Ratio and Additive Term. We say that $\textsc{Alg}$ is c-competitive if there exists β such that

$$\textsc{Alg}(E) \geqslant \textsc{Opt}(E) \,/\, c - \beta. \tag{1}$$

While β can be a function of n, it cannot depend on the input sequence E. For randomized algorithms, we replace $\textsc{Alg}(E)$ with its expected value taken over the random choices of the algorithm.

We note that the use of the additive constant β is typical in the literature on online algorithms [8] for problems where the value of $\textsc{Opt}$ can be arbitrarily large. The additive term is unavoidable for the DSC problem: setting $\beta = 0$ makes the competitive ratio of any deterministic algorithm to be $\Omega(n)$ [21, 15]. On the positive side, the parameter β used in our analysis is merely $1/4$.

Randomized Algorithms. The currently best randomized algorithm was given by Emek et al. [15]; it achieves the competitive ratio of $O(\log^2 n)$. To understand its main idea, consider a straightforward adaptation of the randomized offline algorithm outlined in Subsection 1.1 to the online setting: assume that to color a hyperedge S, it chooses a random color from the palette of $\Theta(\delta^* / \log n)$ colors, where δ^* is *the current* minimum degree of the hypergraph.

This idea would fail because there may be many nodes in S whose current degree is much larger than γ^*, and such nodes may already have many or all of the colors from the palette. Such an algorithm would then make a slow progress in coloring these nodes. To alleviate this problem, an algorithm can use an infinite number of disjoint palettes, indexed by integers, with palette k containing $\Theta(2^k / \log n)$ colors. To color S, an algorithm first randomly chooses a palette index k from $\{\lceil \log \gamma^* \rceil, \dots, \lceil \log \gamma^* \rceil + \log n\}$, and then uses a color randomly picked from the palette k. While the randomized algorithm of Emek et al. [15] is more sophisticated, it builds on this idea, losing the first factor of $O(\log n)$ in palette selection, and the second factor of $O(\log n)$ in the same place as the offline algorithm does.

The best (randomized) lower bound is $\Omega(\log n / \log\log n)$ [15]; it improves on an earlier bound of $\Omega(\sqrt{\log n})$ by Pananjady et al. [21].

Deterministic Algorithms. The deterministic case is much less understood: while the randomized lower bound of $\Omega(\log n / \log\log n)$ holds for deterministic algorithms, the best upper bound is $O(n)$ and is achieved by a simple greedy algorithm [21]. The known results are summarized in Table 1.

1.3 Our Results and Technical Challenges

In our paper, we present a deterministic online algorithm DET that is $O(\log^2 n)$-competitive, an exponential improvement over the previous bound of $O(n)$. To some extent, our algorithm can be seen as a derandomization of an algorithm by Emek et al. [15].

General Derandomization Tools. Unlike approximation algorithms, the derandomization is extremely rare in online algorithms.[3] To understand the difficulty, consider the standard (offline) method of conditional probabilities [3] applied to the offline DSC problem: the resulting algorithm considers all the random choices (color assignments) it could make for a given hyperedge S, and computes the probability that future random choices, *conditioned on the current one*, will lead to the desired solution. Choosing the color that maximizes this probability ensures that the probability of reaching the desired solution does not decrease as subsequent hyperedges are processed. In some cases, exact computations are not possible, but the algorithm can estimate this probability instead by computing the amount called *pessimistic estimator* [23]). This is infeasible in the online setting, since an algorithm does not know the future sets, and thus cannot even estimate these probabilities.[4]

Online-Computable Potential Functions. In the realm of online algorithms, the method of conditional probabilities has been successfully replaced several times by an online-computable potential function that guides the choice of the deterministic algorithm [2, 5, 11, 19].[5]

To illustrate this concept, let c_i be the number of colors that node i gathered so far. We use $\deg(i)$ to denote the *current* degree of node i. The ideas from previous papers would then suggest choosing a potential function of the form $\Phi = \sum_{i \in [n]} \exp(z_i)$, where $z_i = \deg(i) - \gamma \cdot c_i$ for some constant γ. Note that Φ relates the resources expended by an algorithm (growth of the degree) to its progress (number of gathered colors). We will show issues with this choice of Φ later, but first we highlight its advantages.

Since initially $z_i = 0$ for each node i, at the beginning we have $\Phi = n$. The core of the method is to show that, for each considered hyperedge S, it is possible to choose its color, so that Φ does not increase. This is usually achieved by the probabilistic argument: by showing that a *random* choice of a color would lead to an *expected* decrease of Φ. This would guarantee that $\Phi \leqslant n$, which would imply that $z_i \leqslant \ln n$ for each i. Hence, c_i could be lower-bounded with respect to $\deg(i)$ *for each* node. If the colors are chosen from a fixed palette (cf. Subsection 1.1), they are used by all nodes, and thus correspond to valid set covers. This process can be seen as a derandomization of the Chernoff bound / high probability argument.

Technical Challenges. As we discuss below, the ideas from the previous works cannot be directly applied to the DSC problem, and we need to introduce several new techniques.

---

[3] Many online problems (e.g., caching [24, 1] or metrical task systems [7, 9, 10]) exhibit provable exponential discrepancy between the performance of the best randomized and deterministic algorithms. For many other problems, the best known deterministic algorithms are quite different (and more complex) than randomized ones.

[4] As observed by Pananjady et al. [21], however, these probabilities can be estimated if an online algorithm knows the final min-degree $\delta(E)$ in advance, which would lead to an $O(\log n)$-competitive algorithm for this semi-offline scenario. As pointed out by Emek et al. [15], this approach relies heavily on the knowledge of $\delta(E)$.

[5] They should not be confused with the ubiquitous "standard" potential functions that link the states of online and offline algorithms and which are used *only in the analysis*.

The first problem is that setting $z_i = \deg(i) - \gamma \cdot c_i$ disregards the fact that the rate of gathering *new* colors (by a random process) decreases as c_i grows. This effect has been studied in the context of the coupon collector's problem [20]. To overcome this issue, we need to replace $\gamma \cdot c_i$ with an appropriate function of c_i. We note that this problem was not present in the previous applications of the potential function method [2, 5, 11, 19]. In their case, the potential function was used to guide deterministic *rounding*: the function Φ directly compared the cost (or gain) of a deterministic algorithm with that of an online fractional solution. Instead, our solution operates directly on the input, without the need to generate a fractional solution first.

The second problem is that the expected decrease in Φ requires that that the values of $\exp(z_i)$ are decreasing (at least on the average), while a simple argument shows that only the variables z_i are decreasing. In the previous papers, the connection between z_i and $\exp(z_i)$ was made by ensuring that the variables z_i are universally upper-bounded: this can be achieved by first dividing them by the value of Opt. An algorithm was then either given an upper bound on Opt (in the case of the throughput maximization of the virtual circuit routing [11]) or Opt was estimated by standard doubling techniques (in the case of the cost minimization for set cover variants [2, 5, 19]). In the latter case, the algorithm was restarted each time the estimate on Opt was doubled.

The DSC problem (which is an unbounded-gain maximization problem) does not fall into any of the above categories. Moreover, the currently best randomized algorithm [15] suggests that instead of focusing on the global optimum (or the current minimum degree), each node should rather gather colors at a rate depending on *its own degree*. To deal with node degrees growing in time, we partition the steps of each node into phases. In each phase, we look at the fraction whose numerator corresponds to the growth of z_i and whose denominator corresponds to the current approximate node degree. The resulting potential for a node (see Subsection 2.2) is a weighted average of these fractions.

1.4 Related Work

An application of the DSC problem arises in a sensor network, where each node corresponds to a monitoring target and each hyperedge corresponds to a sensor that can monitor a set of targets. At any given time, all targets need to be monitored. A possible battery-saving strategy is to partition the sensors into disjoint groups (each group covering all targets) and activate only one group at a time, while the other sensors remain in a low-power mode [13].

If the assumption that each sensor can participate only in a single group is dropped, this leads to more general sensor coverage problems. The goal is then to maximize the lifetime of the network of sensors while maintaining the coverage of all targets. The offline variants of this problem have been studied both in general graphs [4, 14, 22] and in geometric settings [12, 17].

Another line of work studied the relationship between the minimum degree $\delta(E)$ and the cover-decomposition number $\text{Opt}(E)$. Trivially, $\delta(E)/\text{Opt}(E) \geqslant 1$. This ratio is constant if G is a graph [18], and it is at most $O(\log n)$ for every hypergraph [6]; the latter bound is asymptotically tight [6]. Interestingly enough, all the known papers on the DSC problem (including ours) relate the number of disjoint set covers to $\delta(E)$. It is an open question whether our estimate of the competitive ratio is tight: it could potentially be improved by relating the gain of an algorithm directly to $\text{Opt}(E)$ instead of $\delta(E)$.

1.5 Preliminaries

An input to our problem is a gradually revealed hypergraph $G = (V, E)$. V consists of n nodes numbered from 1 to n, i.e., $V = [n]$. The set E of hyperedges is presented one by one: in a step t, an algorithm learns $S_t \in E$, where $S_t \subseteq [n]$, and has to color it. We say that node i gathers color r if i is incident to a hyperedge colored with r. We say that a color r is *fully used* if all nodes have gathered it. The objective of an algorithm is to maximize the number of fully used colors.

At any point in time, the *degree* of a node j is the current number of hyperedges containing j. Let $\delta(E) = \min_{i \in [n]} |\{S_t \in E : v_i \in S_t\}|$, i.e., $\delta(E)$ is the minimum degree of G at the end of the input E. Clearly, $\textsc{Opt}(E) \leqslant \delta(E)$.

2 Our Deterministic Solution

We first describe our phase-based framework (cf. Subsection 2.1). Then, we introduce the potential function that guides the choice of colors (cf. Subsection 2.2).

2.1 Partitioning Steps into Phases

We start with some notions, defined for each integer $k \geqslant 1$:

$$b_k \triangleq 2^{k-1}, \qquad B_k \triangleq \sum_{j=1}^{k} b_j = 2^k - 1,$$

$$t_k \triangleq \left\lceil \left(1 - \frac{1}{2n}\right) \cdot b_k \right\rceil, \qquad R_k \triangleq \{B_{k-1} + 1, \ldots, B_k\}.$$

Note that $|R_k| = b_k$.

For each hyperedge $S \in E$, $\textsc{Det}$ chooses a color r: this causes all nodes from S to gather color r. However, in our analysis (and even in the algorithm description), we ignore some of these colors.

For each node i independently, $\textsc{Det}$ maintains its *phase* $p(i)$, initially set to 1. In each step, the sole focus of the node i is to gather colors from the *palette* $R_{p(i)}$. We use c_{ik} to denote the number of colors from palette R_k that node i has gathered so far, while it has been *within phase* k. That is, for a node i, we increment the counter $c_{i,p(i)}$ when node i gathers a new color from $R_{p(i)}$ and we never change the counter c_{ik} for $k \neq p(i)$. Clearly, c_{ik} is a lower bound on the number of colors from palette R_k gathered by node i so far.

A phase k for node i terminates when it has gathered t_k colors from palette R_k. In such case, node i increments its phase number $p(i)$ in the next step. Note that t_k is slightly smaller than $|R_k| = b_k$; as we argue later this helps us speed up the process without changing the asymptotic number of fully used colors.

For each node i and an integer $k \geqslant 1$, we maintain two counters w_{ik} and w^*_{ik}; all these counters are initially set to zero. Whenever a new hyperedge containing node i arrives, we increment either $w_{i,p(i)}$ or $w^*_{i,p(i)}$ (we describe this choice later).

▶ **Observation 1.** *$\sum_{k \geqslant 1} (w_{ik} + w^*_{ik})$ is always equal to the current degree of node i. Furthermore, $c_{ik} = w_{ik} = w^*_{ik} = 0$ for every node i and $k > p(i)$.*

2.2 Choosing Colors with Potential Function

We now introduce further notions, defined for each integer $k \geqslant 1$:

$$h \triangleq \lceil \log n \rceil,$$

$$\alpha_k(j) \triangleq \frac{1}{h} \cdot \frac{b_k - j + 1}{b_k}, \qquad \text{(defined for every } j\text{)}$$

$$d_k(\ell) \triangleq \sum_{j=1}^{\ell} \frac{1}{\alpha_k(j)}, \qquad \text{(defined for every } \ell \leqslant b_k\text{)}$$

Finally, for each node i and each integer $k \geqslant 1$, we define

$$z_{ik} \triangleq w_{ik} - 2 \cdot d_k(c_{ik}).$$

Note that since $c_{ik} \leqslant t_k \leqslant b_k$, the value of $d_k(c_{ik})$ is always well defined.

Recall that, for each node i and within phase k, the growth of its degree is $w_{ik} + w^*_{ik}$, while c_{ik} is the number of colors gathered from the palette R_k. Thus, neglecting w^*_{ik}, the value of z_{ik} relates these amounts. As mentioned in Subsection 1.3, a direct comparison of w_{ik} and c_{ik} makes no sense. So instead of c_{ik} we thus use $d_k(c_{ik})$, which corresponds to the expected number of steps in the random process of collecting colors from the palette R_k until c_{ik} distinct colors are found.

Negative values of z_{ik} indicate that node i is gathering colors in phase k faster than the benchmark, while positive values indicate that it is falling behind. Our algorithm will choose a color in such a way that z_{ik}'s are appropriately upper-bounded. To this end, we define the potential for each node i as

$$\Phi_i \triangleq \exp\left(\sum_{k \geqslant 1} \frac{z_{ik}}{4h \cdot b_k}\right),$$

and the total potential as

$$\Phi \triangleq \sum_{i \in [n]} \Phi_i.$$

Now describe the behavior of Det in a single step, when a hyperedge S appears. Let $p(i)$ denote the phase of node i before S appears and let $p_S = \min_{i \in S} p(i)$. For each node $i \in S$,

- Det increments $w_{i,p(i)}$ if $i \leqslant p_S + h - 1$, and
- Det increments $w^*_{i,p(i)}$ if $i \geqslant p_S + h$.

To color S, Det takes the color from the set

$$\biguplus_{p_S \leqslant k \leqslant p_S + h - 1} R_k$$

resulting in a minimum value of potential Φ.

Since the number of colors to consider is bounded by a polynomial in n and the number of sets in the input, Det can be implemented in polynomial time.

3 Analysis Roadmap

To compute the competitive ratio of Det, we need to link Det(E) to $\delta(E)$ for each input E. We start by estimating the former.

▶ **Lemma 2.** *If each node has completed at least ℓ phases, then* $\textsc{Det}(E) \geqslant b_\ell/2$.

Proof. In phase ℓ, each node gathers at least t_ℓ colors from the palette R_ℓ, i.e., all colors from R_ℓ except at most $|R_\ell| - t_\ell \leqslant b_\ell/(2n)$ colors. So all nodes share at least $|R_\ell| - n \cdot (b_\ell/(2n)) \geqslant b_\ell/2$ common colors from R_ℓ, i.e., Det fully uses at least $b_\ell/2$ colors. ◀

Note that we could sum the above bound over all phases completed by all nodes, but this would not change the asymptotic gain of Det.

Upper Bound on Node Degrees. The hard part of our analysis is to appropriately upper-bound the values of w_{ik} and w^*_{ik}, i.e., to show the following two lemmas.

▶ **Lemma 3.** *Fix a node i and an integer $\ell \geqslant 1$. At any time, it holds that* $\sum_{k=1}^{\ell} w_{ik} \leqslant 8h \cdot b_\ell \cdot \ln(4\mathrm{e} \cdot n)$.

▶ **Lemma 4.** *Fix a node i and an integer $\ell \geqslant 1$. At any time, it holds that* $\sum_{k=1}^{\ell} w^*_{ik} \leqslant 16h \cdot b_\ell \cdot \ln(4\mathrm{e} \cdot n)$.

As we already mentioned in Subsection 1.3, to estimate the growth of w_{ik} (Lemma 3), we show that we can always control the total potential. A simple computation shows that the initial potential is equal to n. Furthermore, using a probabilistic method, we can show that Det can always choose a color for a given hyperedge, so that the potential always remains at most n (cf. Section 4). The bound on Φ creates a connection between b_k and z_{ik}, and thus also between b_k and w_{ik}, which allows us to prove Lemma 3 (cf. Subsection 5.1).

The growth of the variables w^*_{ik} is not controlled by the potential, but by the definition of Det, they grow only for nodes whose degree is very high compared to the current minimum degree. By constructing an appropriate charging argument, we can link their growth to the growth of other variables w_{ik} (cf. Subsection 5.2).

Competitive Ratio. Using the three lemmas above, we can relate the final minimum node degree $\delta(E)$ to the number of phases each node has completed. This directly gives the competitive ratio of Det.

▶ **Lemma 5.** *Fix an integer $\ell \geqslant 1$. If $\delta(E) > 24h \cdot \ln(4\mathrm{e} \cdot n) \cdot b_\ell$, then each node has completed at least ℓ phases.*

Proof. Suppose for a contradiction that there exists a node i for which $p(i) \leqslant \ell$ at the input end. We would then have

$$\delta(E) \leqslant \sum_{k \geqslant 1} (w_{ik} + w^*_{ik}) = \sum_{k=1}^{\ell} (w_{ik} + w^*_{ik}) \leqslant 24h \cdot \ln(4\mathrm{e} \cdot n) \cdot b_\ell,$$

where the first two relations follow by Observation 1 and the last one by Lemma 3 and Lemma 4. This would contradict the assumption of the lemma. ◀

▶ **Theorem 6.** *Det is $O(\log^2 n)$-competitive*

Proof. Fix an input sequence E and let $q \triangleq 24h \cdot \ln(4\mathrm{e} \cdot n)$. We consider two cases.

- First, assume that $\delta(E) > q = q \cdot b_1$. Then we can find an integer $\ell \geqslant 1$ such that $q \cdot b_\ell < \delta(E) \leqslant q \cdot b_{\ell+1}$. By Lemma 5, each node then completes at least ℓ phases, and so by Lemma 2, $\textsc{Det}(E) \geqslant b_\ell/2 = b_{\ell+1}/4 \geqslant \delta(E)/(4 \cdot q)$.
- Second, assume that $\delta(E) \leqslant q$. Trivially, $\textsc{Det}(E) \geqslant 0 \geqslant (\delta(E) - q)/(4q)$.

In both cases, $\text{Det}(E) \geqslant (\delta(E) - q)/(4q) \geqslant \text{Opt}(E)/(4q) - 1/4$. i.e., the competitive ratio of Det is at most $4 \cdot q = O(\log^2 n)$.

Note that, by the lower bounds of [21, 15], an additive term (equal to $1/4$ in our case) is inevitable for obtaining a sub-linear competitive ratio. ◀

4 Controlling the Potential

In this section, we show that the potential Φ is always at most n. A simple argument estimates its initial value.

▶ **Lemma 7.** *Initially, the potential Φ is equal to n.*

Proof. Initially, $w_{ik} = c_{ik} = 0$ for every node i and every integer $k \geqslant 1$. Thus, $z_{ik} = 0$, which implies $\Phi_i = \mathrm{e}^0 = 1$. So at the beginning, $\Phi = \sum_{i=1}^n \Phi_i = n$. ◀

Random Experiment. In this section, we use the probabilistic method to argue that there always exists a color for the considered hyperedge, such that the resulting potential does not increase. Since Det always chooses a color that minimizes the potential, this will imply that Φ is always at most n.

Recall how Det chooses a color for a hyperedge S: it sets $p_S = \min_{i \in S} p(i)$, where $p(i)$ is the phase of node i before S appears. Next, it takes the color from the set

$$\mathcal{R} \triangleq \biguplus_{p_S \leqslant k \leqslant p_S + h - 1} R_k$$

which results in a minimum value of the potential Φ.

We replace the above procedure with a random choice of a color from $\mathcal{R}$ and show that the resulting potential does not increase *in expectation*. By the probabilistic method, this will imply that *there exists a deterministic choice* of the color from $\mathcal{R}$ that does not increase the potential.

To this end, we first choose a random integer k uniformly from the set $\{p_S, p_S+1, \dots, p_S + h - 1\}$. Second, conditioned on the first choice, we choose a random color uniformly from the set R_k.

Throughout this section, we focus on a single step, in which the hyperedge S is processed. We use w_{ik}, c_{ik}, z_{ik}, and Φ_{ik} to denote the values of the corresponding variables before S appears. We use w'_{ik} to denote their values immediately after S appears (these variables depend only on S and not on our random experiment). Furthermore, we use c'_{ik}, z'_{ik}, and Φ'_{ik} to denote the *random variables* corresponding to variables right after S is randomly colored.

Whenever we write a relation involving a random variable and not its expectation (e.g., $z'_{ik} \leqslant z_{ik}$), we mean that it holds for all realizations of that random variable.

Analyzing Variables z.

▶ **Lemma 8.** *Fix a node i and an integer $k \geqslant 1$. If $i \notin S$ or $k \neq p(i)$, then $z'_{ik} = z_{ik}$.*

Proof. In this case, $w'_{ik} = w_{ik}$ and $c'_{ik} = c_{ik}$ by the definition of Det (it modifies only variables $w_{i,p(i)}$ and $c_{i,p(i)}$ and only for $i \in S$). The lemma follows by the definition of z_{ik}. ◀

▶ **Lemma 9.** *Fix a node $i \in S$ and let $p = p(i)$. If $p \geqslant p_S + h$, then $z'_{ip} = z_{ip}$.*

Proof. In the considered case, Det increments w^*_{ip} while leaving w_{ip} intact, i.e., $w'_{ip} = w_{ip}$. On the other hand, the random process never picks a color from R_p, and thus $c'_{ip} = c_{ip}$. The lemma follows by the definition of z_{ip}. ◀

▶ **Lemma 10.** *Fix a node $i \in S$ and let $p = p(i)$. If $p \leqslant p_S + h - 1$, then*

$$z'_{ip} = z_{ip} + 1 + \begin{cases} -2/\alpha_p(c_{ip}+1) & \text{with probability } \alpha_p(c_{ip}+1), \\ 0 & \text{otherwise.} \end{cases}$$

Proof. By the definition of p_S, we have $p \geqslant p_S$. Combining this with the lemma assumption, we get $p \in \{p_S, \ldots, p_S + h - 1\}$.

First, we analyze the relation between c'_{ip} and c_{ip}. Note that $c'_{ip} = c_{ip} + 1$ if node i gathers a new color from R_p, and $c'_{ip} = c_{ip}$ otherwise. For a node i to gather a new color from R_p, first the integer chosen randomly from the set $\{p_S, \ldots, p_S + h - 1\}$ must be equal to p (which happens with probability $1/h$). Second, conditioned on the former event, a color chosen randomly from the palette R_p must be different from all c_{ip} colors from R_p gathered so far by node i (which happens with with the probability $(|R_p| - c_{ip})/|R_p| = (b_p - c_{ip})/b_p$). In summary, the probability of gathering a new color is $(1/h) \cdot (b_p - c_{ip})/b_p = \alpha_p(c_{ip} + 1)$. Hence,

$$c'_{ip} = \begin{cases} c_{ip}+1 & \text{with probability } \alpha_p(c_{ip}+1), \\ c_{ip} & \text{otherwise.} \end{cases} \tag{2}$$

Recall that $d_k(\ell) = \sum_{j=1}^{\ell} 1/\alpha_k(j)$. Therefore,

$$d_p(c'_{ip}) - d_p(c_{ip}) = \begin{cases} 1/\alpha_p(c_{ip}+1) & \text{with probability } \alpha_p(c_{ip}+1), \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Finally, in the considered case, $w'_{ip} = w_{ip} + 1$. Since $z_{ik} = w_{ik} - 2 \cdot d_k(c_{ik})$,

$$\begin{aligned} z'_{ip} - z_{ip} &= \left(w'_{ip} - w_{ip}\right) - 2 \cdot \left(d_p(c'_{ip}) - d_p(c_{ip})\right) \\ &= 1 + \begin{cases} -2/\alpha_p(c_{ip}+1) & \text{with probability } \alpha_p(c_{ip}+1), \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$ ◀

For the next lemma, we need the following technical bound (proven in Appendix A).

▷ Claim 11. Fix $\varepsilon \in [0, 1]$, $\alpha \in [\varepsilon, 1]$ and a real x. Let X be a random variable such that

$$X = \begin{cases} x - \varepsilon/\alpha & \text{with probability } \alpha, \\ x & \text{otherwise.} \end{cases}$$

Then, $\mathbb{E}[\mathrm{e}^X] \leqslant \mathrm{e}^{x - \varepsilon/2}$.

▶ **Lemma 12.** *Fix a node $i \in S$ and let $p = p(i)$. Then,*

$$\mathbb{E}\left[\exp\left(\frac{z'_{ip}}{4h \cdot b_p}\right)\right] \leqslant \exp\left(\frac{z_{ip}}{4h \cdot b_p}\right).$$

Proof. If $p \geqslant p_S + h$, then $z'_{ip} = z_{ip}$ by Lemma 9, and the lemma follows trivially. So in the following, we assume that $p \leqslant p_S + h - 1$.

Taking the guarantee of Lemma 10 and dividing its both sides by $4h \cdot b_p$, we get

$$\frac{z'_{ip}}{4h \cdot b_p} = \frac{z_{ip}+1}{4h \cdot b_p} + \begin{cases} -1/(2h \cdot b_p \cdot \alpha) & \text{with probability } \alpha, \\ 0 & \text{otherwise.} \end{cases}$$

where $\alpha \triangleq \alpha_p(c_{ip}+1)$.

Note that $c_{ip} \leqslant t_p - 1 \leqslant b_p - 1$ as otherwise phase p of node i would have ended in an earlier step. Hence,

$$\alpha = \frac{1}{h} \cdot \frac{b_p - c_{ip}}{b_p} \geqslant \frac{1}{h \cdot b_p}.$$

Thus, $X = z'_{ip}/(4h \cdot b_p)$, $x = (z_{ip}+1)/(4h \cdot b_p)$, $\varepsilon = 1/(2h \cdot b_p)$, and α satisfy the conditions of Claim 11. (In particular, $\varepsilon \leqslant \alpha \leqslant 1$.) This claim now yields

$$\mathbb{E}\left[\exp\left(\frac{z'_{ip}}{4h \cdot b_p}\right)\right] \leqslant \exp\left(\frac{z_{ip}+1}{4h \cdot b_p} - \frac{1}{2} \cdot \frac{1}{2h \cdot b_p}\right) = \exp\left(\frac{z_{ip}}{4h \cdot b_p}\right). \qquad ◀$$

Potential Monotonicity.

▶ **Lemma 13.** *For each node i, it holds that $\mathbb{E}[\Phi'_i] \leqslant \Phi_i$.*

Proof. First, assume that $i \notin S$, where S is the considered hyperedge. By Lemma 8, $z'_{ik} = z_{ik}$ for every integer $k \geqslant 1$. In such case, $\Phi'_i = \Phi_i$ (for all outcomes of the random process in the considered step), which trivially implies $\mathbb{E}[\Phi'_i] = \Phi_i$.

Hence, in the following, we assume that $i \in S$. For each integer $k \geqslant 1$, we relate z'_{ik} to z_{ik}. Let $p = p(i)$ be the current phase of node i (before processing hyperedge S). By Lemma 8,

$$z'_{ik} = z_{ik} \qquad \text{for every } k \neq p. \tag{4}$$

Note that (4) holds not only in expectation, but for all realizations of the random variable z'_{ik}. Using the definition of Φ'_i and applying (4) yields

$$\begin{aligned}\Phi'_i &= \exp\left(\frac{z'_{ip}}{4h \cdot b_p} + \sum_{k \neq p} \frac{z'_{ik}}{4h \cdot b_k}\right) = \exp\left(\frac{z'_{ip}}{4h \cdot b_p} + \sum_{k \neq p} \frac{z_{ik}}{4h \cdot b_k}\right) \\ &= \exp\left(\frac{z'_{ip}}{4h \cdot b_p}\right) \cdot \exp\left(\sum_{k \neq p} \frac{z_{ik}}{4h \cdot b_k}\right)\end{aligned}$$

In the last line above, the former multiplier is a random variable, while the latter one is a deterministic value. We can now apply expectation to both sides, which gives us

$$\begin{aligned}\mathbb{E}[\Phi'_i] &= \mathbb{E}\left[\exp\left(\frac{z'_{ip}}{4h \cdot b_p}\right)\right] \cdot \exp\left(\sum_{k \neq p} \frac{z_{ik}}{4h \cdot b_k}\right) \\ &\leqslant \exp\left(\frac{z_{ip}}{4h \cdot b_p}\right) \cdot \exp\left(\sum_{k \neq p} \frac{z_{ik}}{4h \cdot b_k}\right) && \text{(by Lemma 12)} \\ &= \Phi_i. && \text{(by the definition of } \Phi_i\text{)}\end{aligned}$$
◀

▶ **Corollary 14.** *At any time, $\Phi \leqslant n$.*

Proof. The proof follows inductively. At the beginning, $\Phi = n$ by Lemma 7. By Lemma 13, if we randomly choose the color for a new hyperedge (using the random process described in this section), it is guaranteed that the potential does not increase in expectation. Thus, there exists a deterministic choice of the color for the new hyperedge that does not increase the potential. ◀

▶ **Corollary 15.** *At any time, and for each node i, $\Phi_i \leqslant n$.*

Proof. By Corollary 14, $\Phi = \sum_{i \in [n]} \Phi_i \leqslant n$. The claim follows because all Φ_i are positive. ◀

5 Upper-Bounding the Degrees

5.1 Estimating Variables w (Proof of Lemma 3)

So far, we have shown that the potential is always at most n. This allows us to control the weighted average of z_{ik}. Recall that $z_{ik} = w_{ik} - 2 \cdot d(c_{ik})$. Thus, to estimate variables w_{ik}, we first need to upper-bound $d(c_{ik})$.

We start with the following technical claim (proven in Appendix A). H_m denotes the m-th harmonic number.

▷ Claim 16. For each $k \geqslant 1$ it holds that $H_{b_k} - H_{b_k - t_k} \leqslant \ln(4e \cdot n)$.

▶ **Lemma 17.** *Fix a node i and an integer $k \geqslant 1$. Then, $w_{ik} \leqslant z_{ik} + 2h \cdot b_k \cdot \ln(4e \cdot n)$.*

Proof. Note that at all times, $c_{ik} \leqslant t_k$. Using the definition of z_{ik} and the monotonicity of the function d_k, we get

$$w_{ik} = z_{ik} + 2 \cdot d_k(c_{ik}) \leqslant z_{ik} + 2 \cdot d_k(t_k). \tag{5}$$

It remains to upper-bound $d_k(t_k)$. Using the definition of d_k,

$$\begin{aligned} d_k(t_k) &= \sum_{j=1}^{t_k} \frac{1}{\alpha_k(j)} = \sum_{j=1}^{t_k} \frac{h \cdot b_k}{b_k - j + 1} = \sum_{j=b_k - t_k + 1}^{b_k} \frac{h \cdot b_k}{j} \\ &= h \cdot b_k \cdot (H_{b_k} - H_{b_k - t_k}) \leqslant h \cdot b_k \cdot \ln(4e \cdot n). \qquad \text{(by Claim 16)} \end{aligned}$$

The lemma follows by plugging the bound on $d_k(t_k)$ above into (5). ◀

Finally, we can prove the main result of this section, restated below for convenience.

▶ **Lemma 3.** *Fix a node i and an integer $\ell \geqslant 1$. At any time, it holds that $\sum_{k=1}^{\ell} w_{ik} \leqslant 8h \cdot b_\ell \cdot \ln(4e \cdot n)$.*

Proof. It is sufficient to show the lemma right after the last increase of $\sum_{k=1}^{\ell} w_{ik}$. All values of the variables in the proof below are taken at this point. By Corollary 15, $\Phi_i \leqslant n$. Using the definition of Φ_i we get

$$\sum_{k \geqslant 1} \frac{z_{ik}}{4h \cdot b_k} = \ln \Phi_i \leqslant \ln n.$$

Note that for every $k > \ell$, we have $w_{ik} = c_{ik} = 0$, and thus also $z_{ik} = 0$. That is,

$$\ln n \geqslant \sum_{k \geqslant 1} \frac{z_{ik}}{4h \cdot b_k} = \sum_{k=1}^{\ell} \frac{z_{ik}}{4h \cdot b_k} \geqslant \frac{1}{4h \cdot b_\ell} \cdot \sum_{k=1}^{\ell} z_{ik}. \tag{6}$$

We can now sum the guarantee of Lemma 17 over all phases from 1 to ℓ and get

$$\begin{aligned} \sum_{k=1}^{\ell} w_{ik} &\leqslant \sum_{k=1}^{\ell} z_{ik} + 2h \cdot \ln(4e \cdot n) \cdot \sum_{k=1}^{\ell} b_k \\ &\leqslant 4h \cdot b_\ell \cdot \ln n + 4h \cdot \ln(4e \cdot n) \cdot b_\ell \qquad \text{(by (6) and the definition of } b_k) \\ &\leqslant 8h \cdot b_\ell \cdot \ln(4e \cdot n). \end{aligned}$$

◀

5.2 Estimating Variables w* (Proof of Lemma 4)

As mentioned in Section 3, we now construct a charging scheme that maps the increments of variables $w^*_{i\ell}$ to the increments of other variables w_{ik}. This allows us to prove Lemma 4.

▶ **Lemma 18.** *At any time, each node i, and each $\ell \geqslant 1$, we have $w^*_{i\ell} \leqslant \sum_{j\in[n]} \sum_{k=1}^{\ell-h} w_{jk}$.*

Proof. By the definition of DET, the value of $w^*_{i\ell}$ is incremented only when $\ell = p(i)$, node i is contained in the considered hyperedge S, and $\ell \geqslant p_S + h$. By the definition of p_S, this means that there exists at least one node $j \in S$ such that $p(j) = p_S$, and thus the value of w_{j,p_S} is also incremented.

We can thus map each increment of $w^*_{i\ell}$ to an increment of w_{jk} for some j and $k \leqslant \ell - h$, and such a mapping is injective. This means that whenever the left hand side of the lemma inequality increases by 1, the right hand side increases by at least 1, concluding the proof. ◀

▶ **Lemma 4.** *Fix a node i and an integer $\ell \geqslant 1$. At any time, it holds that $\sum_{k=1}^{\ell} w^*_{ik} \leqslant 16h \cdot b_\ell \cdot \ln(4\mathrm{e} \cdot n)$.*

Proof. Fix any integer ℓ'. Using Lemma 18, we get

$$\begin{aligned} w^*_{i\ell'} \leqslant \sum_{j\in[n]} \sum_{k=1}^{\ell'-h} w_{jk} &\leqslant n \cdot 8h \cdot b_{\ell'-h} \cdot \ln(4\mathrm{e} \cdot n) && \text{(by Lemma 3)} \\ &\leqslant 8h \cdot b_{\ell'} \cdot \ln(4\mathrm{e} \cdot n). && \text{(as } b_k = 2^k \text{ and } h = \lceil \log n \rceil) \end{aligned}$$

Hence, $\sum_{k=1}^{\ell} w^*_{ik} \leqslant 16h \cdot b_\ell \cdot \ln(4\mathrm{e} \cdot n)$. ◀

6 Conclusions

In this paper, we have constructed a deterministic $O(\log^2 n)$-competitive algorithm for the DSC problem. Closing the remaining logarithmic gap between the current upper and lower bounds is an interesting open problem, which seems to require a new algorithm that goes beyond the phase-based approach of our and previous work.

We developed new derandomization techniques that extend the existing potential function methods. In particular, we tackled the combination of concentration bound and coupon collector's arguments used in the solution of Emek et al. [15]. We hope that these extensions will be useful for derandomizing other randomized online algorithms, and eventually for providing a coherent online derandomization toolbox.

References

1 Anna Adamaszek, Artur Czumaj, Matthias Englert, and Harald Räcke. An $O(\log k)$-competitive algorithm for generalized caching. *ACM Transactions on Algorithms*, 15(1):6:1–6:18, 2019. doi:10.1145/3280826.

2 Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. The online set cover problem. *SIAM Journal on Computing*, 39(2):361–370, 2009. doi:10.1137/060661946.

3 Noga Alon and Joel H. Spencer. *The Probabilistic Method, Second Edition*. John Wiley, 2000. doi:10.1002/0471722154.

4 Piotr Berman, Gruia Călinescu, C. Shah, and Alexander Zelikovsky. Power efficient monitoring management in sensor networks. In *2004 IEEE Wireless Communications and Networking Conference*, pages 2329–2334. IEEE, 2004. doi:10.1109/WCNC.2004.1311452.

5 Marcin Bienkowski, Björn Feldkord, and Pawel Schmidt. A nearly optimal deterministic online algorithm for non-metric facility location. In *Proc. 38th Symp. on Theoretical Aspects of Computer Science (STACS)*, LIPIcs, pages 14:1–14:17, 2021. doi:10.4230/LIPIcs.STACS.2021.14.

6 Béla Bollobás, David Pritchard, Thomas Rothvoß, and Alex D. Scott. Cover-decomposition and polychromatic numbers. *SIAM Journal on Discrete Mathematics*, 27(1):240–256, 2013. doi:10.1137/110856332.

7 Alan Borodin, Nati Linial, and Michael E. Saks. An optimal on-line algorithm for metrical task system. *Journal of the ACM*, 39(4):745–763, 1992. doi:10.1145/146585.146588.

8 Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.

9 Sébastien Bubeck, Christian Coester, and Yuval Rabani. The randomized k-server conjecture is false! In *Proc. 55th ACM Symp. on Theory of Computing (STOC)*, pages 581–594. ACM, 2023. doi:10.1145/3564246.3585132.

10 Sébastien Bubeck, Michael B. Cohen, James R. Lee, and Yin Tat Lee. Metrical task systems on trees via mirror descent and unfair gluing. In *Proc. 30th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 89–97. SIAM, 2019. doi:10.1137/1.9781611975482.6.

11 Niv Buchbinder and Joseph (Seffi) Naor. Online primal-dual algorithms for covering and packing. *Mathematics of Operations Research*, 34(2):270–286, 2009. doi:10.1287/MOOR.1080.0363.

12 Adam L. Buchsbaum, Alon Efrat, Shaili Jain, Suresh Venkatasubramanian, and Ke Yi. Restricted strip covering and the sensor cover problem. In *Proc. 18th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1056–1063. SIAM, 2007. doi:10.5555/1283383.1283497.

13 Mihaela Cardei and Ding-Zhu Du. Improving wireless sensor network lifetime through power aware organization. *Wireless Networks*, 11(3):333–340, 2005. doi:10.1007/S11276-005-6615-6.

14 Mihaela Cardei, My T. Thai, Yingshu Li, and Weili Wu. Energy-efficient target coverage in wireless sensor networks. In *Proc. 2005 IEEE Int. Conf. on Computer Communications (INFOCOM)*, pages 1976–1984. IEEE, 2005. doi:10.1109/INFCOM.2005.1498475.

15 Yuval Emek, Adam Goldbraikh, and Erez Kantor. Online disjoint set cover without prior knowledge. In *Proc. 27th European Symp. on Algorithms (ESA)*, LIPIcs, pages 44:1–44:16, 2019. doi:10.4230/LIPICS.ESA.2019.44.

16 Uriel Feige, Magnús M. Halldórsson, Guy Kortsarz, and Aravind Srinivasan. Approximating the domatic number. *SIAM Journal on Computing*, 32(1):172–195, 2002. doi:10.1137/S0097539700380754.

17 Matt Gibson and Kasturi R. Varadarajan. Optimally decomposing coverings with translates of a convex polygon. *Discret. Comput. Geom.*, 46(2):313–333, 2011. doi:10.1007/S00454-011-9353-9.

18 Ram P. Gupta. On the chromatic index and the cover index of a multigraph. In *Theory and Applications of Graphs*, pages 204–215. Springer Berlin Heidelberg, 1978.

19 Ngoc Mai Le, Seeun William Umboh, and Ningyuan Xie. The power of clairvoyance for multi-level aggregation and set cover with delay. In *Proc. 2023 ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1594–1610. SIAM, 2023. doi:10.1137/1.9781611977554.CH59.

20 Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis.* Cambridge University Press, USA, 2nd edition, 2017.

21 Ashwin Pananjady, Vivek Kumar Bagaria, and Rahul Vaze. The online disjoint set cover problem and its applications. In *Proc. 2015 IEEE Int. Conf. on Computer Communications (INFOCOM)*, pages 1221–1229. IEEE, 2015. doi:10.1109/INFOCOM.2015.7218497.

22 Ashwin Pananjady, Vivek Kumar Bagaria, and Rahul Vaze. Optimally approximating the coverage lifetime of wireless sensor networks. *IEEE/ACM Transactions on Networking*, 25(1):98–111, 2017. doi:10.1109/TNET.2016.2574563.

23 Prabhakar Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *Journal of Computer and System Sciences*, 37(2):130–143, 1988. doi:10.1016/0022-0000(88)90003-7.

24 Neal E. Young. On-line file caching. *Algorithmica*, 33(3):371–383, 2002. doi:10.1007/S00453-001-0124-5.

A Proofs of Technical Claims

▷ Claim 11. Fix $\varepsilon \in [0,1]$, $\alpha \in [\varepsilon, 1]$ and a real x. Let X be a random variable such that

$$X = \begin{cases} x - \varepsilon/\alpha & \text{with probability } \alpha, \\ x & \text{otherwise.} \end{cases}$$

Then, $\mathbb{E}[\mathrm{e}^X] \leqslant \mathrm{e}^{x-\varepsilon/2}$.

Proof. Using $\alpha \geqslant \varepsilon$, we obtain

$$(1 + \varepsilon/\alpha) \cdot (1 - \varepsilon/(2\alpha)) = 1 + \varepsilon/(2\alpha) - \varepsilon^2/(2\alpha^2) \geqslant 1. \tag{7}$$

Next, we use the relation $\mathrm{e}^{-y} \leqslant (1+y)^{-1}$ (holding for every $y > -1$) and (7) to argue that

$$\exp(-\varepsilon/\alpha) \leqslant (1 + \varepsilon/\alpha)^{-1} \leqslant 1 - \varepsilon/(2\alpha). \tag{8}$$

Finally, this gives

$$\begin{aligned}
\mathbb{E}[\mathrm{e}^X] &= \alpha \cdot \exp(x - \varepsilon/\alpha) + (1-\alpha) \cdot \exp(x) \\
&= \mathrm{e}^x \cdot (\alpha \cdot \exp(-\varepsilon/\alpha) + 1 - \alpha) \\
&\leqslant \mathrm{e}^x \cdot (\alpha - \varepsilon/2 + 1 - \alpha) && \text{(by (8))} \\
&\leqslant \mathrm{e}^{x-\varepsilon/2}. && \text{(as } 1 - \varepsilon/2 \leqslant \mathrm{e}^{-\varepsilon/2} \text{ for each } \varepsilon\text{)}
\end{aligned}$$

◀

▷ Claim 16. For each $k \geqslant 1$ it holds that $H_{b_k} - H_{b_k - t_k} \leqslant \ln(4\mathrm{e} \cdot n)$.

Proof. First, we note that $\ln m \leqslant H_m \leqslant 1 + \ln m = \ln(\mathrm{e} \cdot m)$ for every $m \geqslant 1$. Thus,

$$H_{b_k} - H_{b_k - t_k} \leqslant \ln(\mathrm{e} \cdot b_k) - \ln(b_k - t_k). \tag{9}$$

If $b_k < 4n$, the claim follows immediately. Otherwise,

$$\begin{aligned}
b_k - t_k &= b_k - \lceil (1 - 1/(2n)) \cdot b_k \rceil && \text{(by the definition of } t_k\text{)} \\
&\geqslant b_k - (1 - 1/(2n)) \cdot b_k - 1 \\
&= b_k/(2n) - 1 \\
&\geqslant b_k/(4n). && \text{(as } b_k \geqslant 4n\text{)}
\end{aligned}$$

Plugging this bound to (9) yields $H_{b_k} - H_{b_k - t_k} \leqslant \ln(\mathrm{e} \cdot b_k) - \ln(b_k/(4n)) = \ln(4\mathrm{e} \cdot n)$. ◀