Efficient Splitting of Necklaces

Noga Alon

Department of Mathematics, Princeton University, Princeton, NJ 08544, USA
Schools of Mathematics and Computer Science, Tel Aviv University, Tel Aviv, Israel

Andrei Graur

Department of Management Science and Engineering, Stanford University, Stanford, CA 94305, USA

Abstract

We provide efficient approximation algorithms for the Necklace Splitting problem. The input consists of a sequence of beads of n types and an integer k. The objective is to split the necklace, with a small number of cuts made between consecutive beads, and distribute the resulting intervals into k collections so that the discrepancy between the shares of any two collections, according to each type, is at most 1. We also consider an approximate version where each collection should contain at least a $(1-\varepsilon)/k$ and at most a $(1+\varepsilon)/k$ fraction of the beads of each type. It is known that there is always a solution making at most $n(k-1)$ cuts, and this number of cuts is optimal in general. The proof is topological and provides no efficient procedure for finding these cuts. It is also known that for $k = 2$, and some fixed positive ε, finding a solution with n cuts is PPAD-hard.

We describe an efficient algorithm that produces an ε-approximate solution for $k = 2$ making $n(2+\log(1/\varepsilon))$ cuts. This is an exponential improvement of a $(1/\varepsilon)^{O(n)}$ bound of Bhatt and Leighton from the 80s. We also present an online algorithm for the problem (in its natural online model), in which the number of cuts made to produce discrepancy at most 1 on each type is $\tilde{O}(m^{2/3}n)$, where m is the maximum number of beads of any type. Lastly, we establish a lower bound showing that for the online setup this is tight up to logarithmic factors. Similar results are obtained for $k > 2$.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases necklace splitting, necklace halving, approximation algorithms, online algorithms, discrepancy

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: https://arxiv.org/abs/2006.16613

Funding *Noga Alon*: Research supported in part by NSF grant DMS-1855464, BSF grant 2018267 and the Simons Foundation.

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:18
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

1.1 The problems

The Necklace Splitting problem deals with a fair partition of a necklace with beads of n colors among k agents. The objective is to cut the necklace into intervals and distribute them to the agents in an equitable way. Before adding more on the background, we give the formal definition of the problem.

▶ **Definition 1.** ***(Necklace Splitting)*** *An instance of Necklace Splitting for n colors and k agents consists of a set of beads ordered along a line, where each bead is colored by a color $i \in [n] = \{1, 2, \dots, n\}$. The goal is to split the necklace, via at most $n(k-1)$ cuts made between consecutive beads, into intervals and distribute them to the k agents so that for each color i, every agent gets either $\lceil \frac{m_i}{k} \rceil$ or $\lfloor \frac{m_i}{k} \rfloor$ beads of color i, where m_i is the number of beads of color i.*

Note that this definition is slightly broader than the one given in [1], where it is assumed that m_i is divisible by k for all $i \in [n]$. However, as shown in [5], these two forms of the Necklace Splitting problem are equivalent. We call the special case $k = 2$ of two agents the Necklace Halving problem. A related problem is the ε-Consensus Splitting problem. Its formal definition is the following:

▶ **Definition 2.** ***(ε-Consensus Splitting)*** *An instance $I_{n,k}$ of ε-Consensus Splitting with n measures and k agents consists of n non-atomic probability measures on the interval $[0, 1]$, which we denote by μ_i, for $i \in [n] = \{1, 2, , \dots, n\}$. The goal is to split the interval, via at most $n(k-1)$ cuts, into subintervals and distribute them to the k agents so that for every two agents $a, b \in [k]$ and every measure $i \in [n]$, we have $|\mu_i(U_a) - \mu_i(U_b)| \le \frac{2\varepsilon}{k}$, where U_a, U_b are the unions of all intervals a, b receive, respectively.*

The ε-Consensus Splitting problem can be viewed as a continuous variant of the Necklace Splitting problem. Furthermore, as will be shown in the proofs of our results, every instance of Necklace Splitting can be converted into an instance of ε-Consensus Splitting. We also consider the **ε-approximate version of Necklace Splitting**, where the goal is to split the necklace so that the difference between the shares of any two agents, according to each type i, is at most $2\varepsilon m_i/k$.

The existence of a solution for the Necklace Splitting problem using at most $n(k-1)$ cuts, a bound which is tight in general, was proved, using topological arguments, first for $k = 2$ agents in [20] (see also [6] for a short proof and [21] for an earlier continuous version), and then for the general case of k agents in [1]. A more recent proof of this existence result appears in [22]. However, the proofs are non-constructive. The Necklace Halving problem is first discussed in [11]. The problem of finding an efficient algorithmic proof of Necklace Splitting is mentioned in [2].

Recently, there have been several results regarding the hardness of the Necklace Halving problem. These are discussed in the next subsection. These suggest pursuing the challenge of finding efficient approximation algorithms, as well as that of proving non-conditional hardness in restricted models.

1.2 Hardness and Approximation

PPA and PPAD are two complexity classes introduced in the seminal paper of Papadimitriou, [23]. Both of these are contained in the class TFNP, which is the complexity class of *total search* problems, consisting of all problems in NP where a solution exists for every instance. A problem is PPA-complete if and only if it is polynomially equivalent to the canonical problem LEAF, described in [23]. Similarly, a problem is PPAD-complete if and only if it is polynomially equivalent to the problem END-OF-THE-LINE. A problem is PPA-hard or PPAD-hard if the respective canonical problem is polynomially reducible to it. A number of important problems, such as several versions of Nash Equilibrium [15] and Market Equilibrium [14], have been proved to be PPAD-complete. It is known that PPAD $\subseteq$ PPA. Hence, PPA-hardness implies PPAD-hardness. Filos-Ratsikas and Goldberg showed that the Necklace Halving problem, as well as the ε-Consensus Halving problem, is PPA-hard [17], see also [18], [16]. Additionally, in [19] it is shown that for a fixed constant $\delta > 0$, and ε inversely polynomial in n, obtaining a solution to the ε-Consensus Halving with fewer than $n + n^{1-\delta}$ is PPA-hard. Our main objective here is to find efficient approximation algorithms for the problems. Although not directly related to our results, it is worth mentioning that in [12] it is proved that for $k = 2$ agents it is NP-hard to minimize the number of cuts for instances where the optimal number is less than n, even with 2 beads of each type.

1.3 Our contribution

We consider approximation algorithms for two versions of the problem, namely the online and the offline versions. We allow the algorithms to make more than n cuts, and expect either a *proper* solution or an ε-approximate one. A *proper* solution is a finite set of cuts and a distribution of the resulting intervals to the k agents so that the absolute discrepancy is at most 1. The absolute discrepancy here and in what follows is the maximum discrepancy, over all types, between the shares of beads of this type allocated to any two agents. An ε-approximate solution is a relaxation in which the discrepancy in any type is at most a fraction $2\varepsilon/k$ of the number of beads of this type. The objective is to minimize the number of cuts the algorithm makes. This problem for the ε-approximate version has been considered earlier in [10] and [13].

In addition to approximation, we also consider hardness in the online model, discussed in the next subsection. In the online model, the hardness is measured by the minimum number of cuts needed to produce a proper solution. Lower bounds on the number of cuts needed in this model provide a barrier for what online algorithms can achieve.

Some of our ideas for finding deterministic approximation algorithms are inspired by papers in Discrepancy Minimization, such as [4], [8], [7] and [9]. In [4], the terminology refers to the **Balancer** as the entity with the designated task of minimizing the absolute discrepancy between agents. We adopt the same terminology here. Thus, the Balancer has the role of an algorithm that makes cuts and assigns the resulting intervals to agents in order to achieve a proper solution for Necklace Splitting.

Our main algorithmic results are summarized in the theorems below. The upper and lower bounds for the number of cuts obtained for the online model appear in the table at the end of this subsection. Throughout the paper, for Necklace Halving, we use the notation $m = \max_{i \in [n]} m_i$ where m_i is the number of beads of color i, and n is the number of types (=colors).

▶ **Theorem 1.** *There exists an efficient, deterministic, offline algorithm that provides a proper solution to the Necklace Halving problem, making at most* $n(\log m + O(1))$ *cuts.*

Here and in what follows an efficient algorithm means an algorithm whose running time is polynomial in the length of the input necklace.

In [10] and [13] the authors describe offline algorithms for the ε-approximate version of Necklace Halving, making $O((\frac{1}{\varepsilon})^{\Theta(n)})$ cuts. Our techniques here provide an algorithm that requires only $n(\log(1/\varepsilon) + O(1))$ cuts for the problem, yielding an exponential improvement for the number of cuts.

▶ **Theorem 2.** *There exists an efficient, deterministic, online algorithm that provides a proper solution to the Necklace Halving problem, making at most* $O(m^{2/3} \cdot n(\log n)^{1/3})$ *cuts.*

The algorithmic results in the online model, and the nearly matching lower bounds we establish appear in the table below. Note that the algorithms are optimal up to constant factors for any fixed constant $n \geq 3$. In the lower bounds for Online Necklace Halving, we always assume that $m_i = m$ for all $i \in [n]$.

Problem	$n = 2$ colors	$n \geq 3,\ n = O(1)$ colors	n colors (general case)
Upper bound	$O(m^{2/3})$	$O(m^{2/3})$	$O(m^{2/3} \cdot n(\log n)^{1/3})$
Lower bound	$\Omega(\sqrt{m})$	$\Omega(m^{2/3})$	$\Omega(n \cdot m^{2/3})$

1.4 Computational model and online version

The offline computational model considered here is natural. The input for Necklace Splitting, for an instance with k agents and n colors, consists of a series of indices, each one taking a value in $[n]$, which represents the color of the respective bead. The runtime is, as usual, the number of basic operations the algorithm makes to provide a solution.

Next, we describe the online model. The parameters k, n and m_i for $i \in [n]$ are given in advance. We refer to *time* t, $0 \leq t \leq \sum_{i\in[n]} m_i - 1$ as the state after the first t beads were revealed and decisions about cutting before any of these have already been made. The beads are revealed one by one in the following way: for integral $t, 0 \leq t \leq \sum_{i\in[n]} m_i - 1$, at time t the Balancer receives the color of bead number $t+1$ and is given the opportunity to make a cut between beads t and $t+1$, where this decision is irreversible. If a cut is made, and J is the newly created interval, the Balancer also has to choose immediately the agent that gets J, before advancing to time $t+1$.

1.5 Techniques

The proofs in the paper combine combinatorial and probabilistic ideas with linear algebra and geometric tools. Theorem 1 is proved by converting the instance of Necklace Halving into a continuous instance, which can be considered an instance of ε-Consensus Halving, where the $[0,1]$ interval is colored by n colors. We reason about finding a solution to this ε-Consensus Halving instance, for a suitable ε, and then adapt the algorithm to obtain a valid solution for the discrete Necklace Halving instance. The algorithm for the continuous instance is based on Carathéodory's Theorem for cones, and involves linear algebra manipulations. To obtain a solution for the discrete instance from the solution to the continuous instance, we describe how to shift cuts at the end to ensure they are not made in the interior of (intervals corresponding to) beads.

The online algorithm discussed in Theorem 2 is inspired by known techniques used in online algorithms for discrepancy minimization. The idea here is to cut the necklace into pieces, each having a sufficiently small number of beads of each color. The problem then becomes an online discrepancy problem, where one can use a derandomization of a natural randomized algorithm that proceeds by using an appropriate potential function motivated by the method of conditional expectations. Obtaining discrepancy ≤ 1 at the end of the necklace traversal requires a modification to the potential function technique, that handles beads of certain colors in a more careful manner once the remaining beads of these colors become scarce. The lower bound showing that the online algorithm is optimal up to logarithmic factors is proved in two steps. The first one is an argument showing that if anytime during the process the discrepancy between the shares allocated so far to the two agents according to one of the colors is relatively large, while according to another color both shares are 0, then a large number of cuts is required to ensure an appropriate solution at the end. In the second step, it is proved that in order to keep the discrepancy according to each color sufficiently small during the process, a large number of cuts is needed. This is shown by introducing and analyzing appropriate potential functions, where the challenge here is to define functions that enable the adversary to ensure they will keep growing for any choice of a place to cut, and any allocation of the resulting interval, provided that the interval created is not too short. One of the lemmas in the proof here is based on the fact that a certain matrix is totally unimodular. The full details appear in the following sections.

1.6 Structure

The structure of the rest of the paper is as follows: in Section 2 we present the approximation algorithm for the offline version of the problem. Section 3 contains the algorithm for the online version. Section 4 contains the lower bounds for the online model. The final Section 5 contains several extensions and open problems. To simplify the presentation we omit all floor and ceiling signs throughout the paper whenever these are not crucial. All logarithms are in base 2, unless otherwise specified.

2 An offline algorithm

Proof of Theorem 1:

Proof. Given a necklace with m_i beads of color i for $1 \leq i \leq n$, where $m = \max m_i$, construct an instance of ε-Consensus Halving as follows. Replace each bead of color i by an interval of i-measure $1/m_i$ and j-measure 0 for all $j \neq i$. These intervals are placed next to each other according to the order in the necklace, and their lengths are chosen so that altogether they cover $[0, 1]$. We first give a marking procedure that splits the continuous necklace so that the absolute discrepancy is at most ε, with $\varepsilon = \frac{1}{2m}$. Then, we show how to modify the solution from the continuous instance to the discrete necklace so that the cuts are made between consecutive beads and we obtain a proper solution.

Given n non-atomic measures μ_i on the interval $[0, 1]$ we describe an efficient algorithm that cuts the interval in at most $n(2 + \lceil \log_2 \frac{1}{\varepsilon} \rceil)$ places and splits the resulting intervals into two collections C_0, C_1 so that $\mu_i(C_j) \in [\frac{1}{2} - \frac{\varepsilon}{2}, \frac{1}{2} + \frac{\varepsilon}{2}]$ for all $i \in [n], 0 \leq j \leq 1$. Note, first, that if the collection C_1 has the right amount according to each of the measures μ_i, so does the collection C_0. For each interval $I \subset [0, 1]$ denote $\mu(I) = \mu_1(I) + \ldots + \mu_n(I)$. Thus $\mu([0, 1]) = n$. Using $2n - 1$ cuts split $[0, 1]$ into $2n$ intervals $I_1, I_2, \ldots, I_{2n}$ so that $\mu(I_r) = 1/2$

for all r. Note that it is easy to find these cuts efficiently, since each measure μ_i is uniform on its support.

For each interval I_r let v_r denote the n-dimensional vector $(\mu_1(I_r), \mu_2(I_r), \ldots, \mu_n(I_r))$.

By a simple linear algebra argument, which is a standard fact about the properties of basic solutions for Linear Programming problems, one can write the vector $(1/2, 1/2, \ldots, 1/2)$ as a linear combination of the vectors v_r with coefficients in $[0, 1]$, where at most n of them are not in $\{0, 1\}$. This follows from Carathédory's Theorem for cones. For completeness, we include the proof, which also shows that one can find coefficients as above efficiently. Start with all coefficients being $1/2$. Call a coefficient which is not in $\{0, 1\}$ *floating* and one in $\{0, 1\}$ *fixed*. Thus at the beginning all $2n$ coefficients are floating. As long as there are more than n floating coefficients, find a nontrivial linear dependence among the corresponding vectors and subtract a scalar multiple of it which keeps all floating coefficients in the closed interval $[0, 1]$ shifting at least one of them to the boundary $\{0, 1\}$, thus fixing it.

This process clearly ends with at most n floating coefficients. The intervals with fixed coefficients with value 1 are now assigned to the collection C_1 and those with coefficient 0 to C_0. The rest of the intervals remain. Split each of the remaining intervals into two intervals, each with μ-value $1/4$. We get a collection $J_1, J_2, \ldots, J_m$ of $m \leq 2n$ intervals, each of them has the coefficient it inherits from its original interval. Each such interval defines an n-vector as before, and the sum of these vectors with the corresponding coefficients (in $(0, 1)$) is exactly what the collection C_1 should still get to have its total vector of measures being $(1/2, \ldots, 1/2)$.

As before, we can shift the coefficients until at most n of them are floating, assign the intervals with $\{0, 1\}$ coefficients to the collections C_0, C_1 and keep at most n intervals with floating coefficients. Split each of those into two intervals of μ-value $1/8$ each and proceed as before, until we get at most n intervals with floating coefficients, where the μ-value of each of them is at most $\varepsilon/2$. This happens after at most $\lceil \log_2(1/\varepsilon) \rceil$ rounds. In the first one, we have made $2n - 1$ cuts and in each additional round at most n cuts. Thus the total number of cuts is at most $n(2 + \lceil \log_2(1/\varepsilon) \rceil) - 1$.

From now on we add no additional cuts, and show how to allocate the remaining intervals to C_0, C_1. Let $\mathcal{I}$ denote the collection of intervals with floating coefficients. Then $|\mathcal{I}| \leq n$ and $\mu(I) \leq \varepsilon/2$ for each $I \in \mathcal{I}$. This means that

$$\sum_{i=1}^{n} \sum_{I \in \mathcal{I}} \mu_i(I) \leq n\varepsilon/2$$

It follows that there is at least one measure μ_i so that

$$\sum_{I \in \mathcal{I}} \mu_i(I) \leq \varepsilon/2.$$

We can think of the remaining floating coefficients as the fraction of each corresponding interval that agent 1 owns. Observe that for any assignment of the intervals $I \in \mathcal{I}$ to the two collections C_0, C_1, the total μ_i measure of C_1 (and hence also of C_0) lies in $[1/2-\varepsilon/2, 1/2+\varepsilon/2]$, as this measure with the floating coefficients is exactly $1/2$ and any allocation of the intervals with the floating coefficients changes this value by at most $\varepsilon/2$. We can thus ignore this measure, for ease of notation assume it is measure number n, and replace each measure vector of the members in $\mathcal{I}$ by a vector of length $n - 1$ corresponding to the other $n - 1$ measures. If $|\mathcal{I}| > n - 1$ (that is, if $|\mathcal{I}| = n$), then it is now possible to shift the floating coefficients as before until at least one of them reaches the boundary, fix it assigning its interval to C_1 or C_0 as needed, and omit the corresponding interval from $\mathcal{I}$ ensuring its size is at most $n - 1$. This means that for the modified $\mathcal{I}$ the sum

$$\sum_{i=1}^{n-1} \sum_{I \in \mathcal{I}} \mu_i(I) \leq (n-1)\varepsilon/2.$$

Hence there is again a measure i, $1 \leq i \leq n-1$ so that

$$\sum_{I \in \mathcal{I}} \mu_i(I) \leq \varepsilon/2.$$

Again, we may assume that $i = n-1$, observe that measure $n-1$ will stay in its desired range for any future allocation of the remaining intervals, and replace the measure vectors by ones of length $n-2$. This process ends with an allocation of all intervals to C_1 and C_0, ensuring that at the end $\mu_i(C_j) \in [1/2 - \varepsilon/2, 1/2 + \varepsilon/2]$ for all $1 \leq i \leq n$, $0 \leq j \leq 1$. These are the desired collections. It is clear that the procedure for generating them is efficient, requiring only basic linear algebra operations.

The intervals separated by the marks are partitioned by the algorithm into two collections forming a solution of the continuous problem. Note that the continuous solution would give discrepancy at most $\max_{i\in[n]} m_i \cdot \varepsilon \leq 1/2$ in terms of beads if we were allowed to cut at the marked points. The only subtle point is that some of the marks may be in the interior of small intervals corresponding to beads, and we wish to cut only between beads.

Call a mark between two consecutive beads *fixed* and call the other marks *floating*. We first show how to shift each of the floating marks so that the absolute discrepancy does not increase beyond 1/2 and all but at most one mark for each color are made between two consecutive beads. To do so, if there exists a floating mark between two intervals assigned to the same agent eliminate it and merge the two intervals. If there is no such mark and there are at least two floating marks in the interior of intervals corresponding to color i, we shift both of them by the same amount in the appropriate way until at least one of them becomes fixed. If during this simultaneous shift one of the two marks arrives in a spot occupied by a different mark, we stop the shift and discard one of the duplicate marks. Note that the quantities the two agents receive do not change.

This procedure reduces the number of floating marks until there is at most one floating mark for each color. If there is such a floating mark, round it to the closest boundary between beads noting that this can increase the absolute discrepancy by at most 1. Therefore, once all marks are fixed, the absolute discrepancy is $\leq 3/2$. Since all the cuts are between consecutive beads, this discrepancy has to be an integer, and thus it is at most 1, as desired. The number of cuts made is $\leq n(2 + \lceil \log_2 \frac{1}{\varepsilon} \rceil) = n(3 + \lceil \log_2 m \rceil) = n(\log m + O(1))$.

◀

Remarks:

- The argument can be extended to splitting into k nearly fair collections of intervals. See section 5 for more details.
- The ε-approximate Necklace Halving problem can be solved with $n(\log(\frac{1}{\varepsilon}) + O(1))$ cuts by using the above algorithm for the continuous instance with the required value of ε.
- The proof can be adapted to obtain a solution with $n(\log(\frac{1}{\varepsilon}) + O(1))$ cuts to the ε-Consensus Halving problem, with the appropriate natural assumptions about the way the measures are presented.
- In [19] the authors give an efficient algorithm for solving a special case of the ε-Consensus Halving problem that works for probability measures each of which is uniform on a single interval. The algorithm provides a solution making at most n cuts for this special case.

3 An online algorithm

Proof of Theorem 2:

Proof. We describe an efficient online algorithm that achieves absolute discrepancy at most 1. The algorithm makes $O(m^{2/3}n(\log n)^{1/3})$ cuts. It is worth mentioning that the main part of the algorithm is a derandomization of a simple randomized algorithm which cuts the necklace into pieces each of which has a sufficiently small number of beads of each color and then assigns them randomly and uniformly to the two agents.

Note first that if, say, $\log n > m/1000$, the result is trivial, as less than nm cuts suffice to split the necklace into single beads, hence we may and will assume that $m \geq 1000 \log n$. Throughout the algorithm we call the beads that have not yet been revealed the *remaining beads*. This definition makes sense as in the online model the beads of the necklace are revealed one by one. We provide a cutting rule and a distribution rule. During the algorithm, we call a color i *critical* if the number of remaining beads of this color is smaller than $20\frac{m_i}{m^{1/3}}(\log n)^{1/3}$, otherwise it is *normal*. When encountering a bead of a critical color i while traversing the necklace, the algorithm makes a cut before and after it, allocating that bead to the agent with a smaller number of beads of this type, where ties are broken arbitrarily. We call such cuts that are made right before or after beads of a critical color *forced*.

In addition to the rule about forced cuts, we provide a rule determining when to stop traversing the necklace and make a cut when no beads of a critical color are seen. Define $g = \frac{100}{8m^{2/3}(\log n)^{1/3}}$, and for every $i \in [n]$, $g_i = m_i g = \frac{100 m_i}{8m^{2/3}(\log n)^{1/3}}$. Whenever after the last cut made after bead number x we reach a bead number y so that $[x, y]$ (the interval containing beads $x+1, x+2, ..., y$) contains at most g_i beads of color i for every i that is normal at that time and exactly g_j beads of some normal color j, we make a cut. As explained above, the exception to this rule is when we encounter a bead of a color i that is critical before the portion following the last cut has enough beads of some normal color. If $g_i \leq 1$ for some color i, then we cut before and after each bead of color i, essentially treating color i as critical from the beginning.

To decide about the allocation of the intervals created we define, for each color $i \in [n]$, a potential function $\phi_i(t)$, and a function $\psi_i(t)$ that is an upper bound of ϕ_i and is computable efficiently. The variable t here will denote, throughout the algorithm, the index of the last cut made.

The functions ϕ_i, ψ_i are defined by considering an appropriate probabilistic process. For each $i \in [n]$, let X_i be the random variable whose value is the difference between the number of beads of color i belonging to agent 1 and that belonging to agent 2 if after each cut the interval created is assigned to a uniform random agent. Let ε_k be 1 if the k'th interval is assigned to agent 1 and -1 otherwise. Therefore $X_i = \sum_{j=1}^{p} \varepsilon_j a_j$, where $p-1$ is the total number of cuts made and a_j the number of beads of color i on interval I_j, the j'th created interval. The distribution defining X_i is the one where each ε_j is 1 or -1 randomly, uniformly and independently. The function $\phi_i(t)$ is defined as follows

$$\phi_i(t) = \mathbb{E}\left[\frac{e^{\lambda X_i/m_i} + e^{-\lambda X_i/m_i}}{2} | \varepsilon_1, \varepsilon_2, ..., \varepsilon_t\right]$$

This is a conditional expectation, where the conditioning is on the allocation of the first t intervals represented by $\varepsilon_1, \ldots, \varepsilon_t$, and where $\lambda = \frac{4m^{1/3}(\log n)^{2/3}}{10}$. (This choice of λ will become clear later). The purpose of the division by m_i is to normalize the exponent of the potential functions to ensure maintaining a relatively small discrepancy for all colors i

simultaneously. Since $X_i = \sum_j \varepsilon_j a_j$, where a_j is the number of beads on the j'th interval of color i, we have that

$$\phi_i(t) = \mathbb{E}\left[\frac{e^{\lambda \sum_j \varepsilon_j a_j/m_i} + e^{-\lambda \sum_j \varepsilon_j a_j/m_i}}{2} | \varepsilon_1, \varepsilon_2, ..., \varepsilon_t\right]$$

The function $\psi_i(t)$ is defined in a way ensuring it upper bounds the function $\phi_i(t)$. It is convenient to split each $\phi_i(t)$ into

$$\frac{1}{2}\mathbb{E}\left[e^{\lambda \sum_j \varepsilon_j a_j/m_i} | \varepsilon_1, .., \varepsilon_t\right] + \frac{1}{2}\mathbb{E}\left[e^{-\lambda \sum_j \varepsilon_j a_j/m_i} | \varepsilon_1, .., \varepsilon_t\right].$$

For simplicity, denote the first term ϕ_i' and the second term ϕ_i''. Therefore

$$\phi_i'(t) = \frac{1}{2}\mathbb{E}\left[e^{\lambda \sum_j \varepsilon_j a_j/m_i} | \varepsilon_1, .., \varepsilon_t\right] = \frac{1}{2} e^{\lambda \sum_{j=1}^{t} \varepsilon_j a_j/m_i} \cdot \prod_{j \geq t+1} \left(\frac{e^{\lambda a_j/m_i} + e^{-\lambda a_j/m_i}}{2}\right)$$

$$= \frac{1}{2} e^{\lambda \sum_{j=1}^{t} \varepsilon_j a_j/m_i} \cdot \prod_{j \geq t+1} \cosh(\lambda a_j/m_i)$$

A similar expression exists for ϕ_i''. Define $s_t = \sum_{j=1}^{t} a_j/m_i$ and $u_t = \sum_{j=1}^{t} \varepsilon_j a_j/m_i$. By the discussion above

$$\phi_i(t) = \frac{e^{\lambda u_t} + e^{-\lambda u_t}}{2} \prod_{j \geq t+1} \cosh(\lambda a_j/m_i).$$

Using the well-known inequality that $\cosh(x) \leq e^{x^2/2}$, it follows that

$$\phi_i(t) \leq \frac{e^{\lambda u_t} + e^{-\lambda u_t}}{2} e^{\lambda^2 \sum_{j \geq t+1} (a_j/m_i)^2/2}.$$

By the way the cuts are produced $a_j \leq g_i$ for all j, and hence

$$\sum_{j=t+1} (a_j/m_i)^2 \leq \max_{j \geq t+1} (|a_j/m_i|) \cdot (\sum_{j \geq t+1} a_j/m_i) \leq g \cdot (\sum_{j \geq t+1} a_j/m_i) = g(1 - s_t).$$

Therefore

$$\phi_i(t) \leq \frac{e^{\lambda u_t} + e^{-\lambda u_t}}{2} e^{\lambda^2 g(1-s_t)/2}.$$

Define $\psi_i(t)$ to be the above upper bound for $\phi_i(t)$, that is

$$\psi_i(t) = \frac{e^{\lambda u_t} + e^{-\lambda u_t}}{2} e^{\lambda^2 g(1-s_t)/2}.$$

Note that $\psi_i(t)$ can be easily computed efficiently at time t, since s_t and u_t (as well as g and λ) are known at this point.

Having defined the potential functions ϕ_i and their upper bounds ψ_i, we are now ready to describe the allocation rule following cuts that create intervals with no beads of any critical color. (The rule for allocating intervals consisting of a single bead of a critical color has already been described). Initialize $\phi(0) = \sum_{i \in [n]} \phi_i(0), \psi(0) = \sum_{i \in [n]} \psi_i(0)$, where by convention $\psi_i(0) = e^{g\lambda^2/2}$. After each cut t during the process, we define $\phi(t) = \sum_{i \text{ normal}} \phi_i(t)$ and

$\psi(t) = \sum_{i \text{ normal}} \psi_i(t)$. In other words, once a color i becomes critical, the terms ϕ_i and ψ_i are dropped from the respective expressions.

Having allocated the first t intervals, at cut $t+1$, we choose ε_{t+1}, which corresponds to a choice of the agent who gets the interval, in order to minimize $\psi(t+1)$. To show that this algorithm produces a proper solution, where the absolute discrepancy at the end is at most 1, we prove the following two claims:

▷ Claim 1. The upper bound $\psi(t)$ is (weakly) decreasing in the variable t.

▷ Claim 2. For each i, after each cut made before the color becomes critical, the discrepancy in color i is at most $10\frac{m_i}{m^{1/3}}(\log n)^{1/3}$ (in absolute value).

Claim 2 implies that after the first cut that causes color i to become critical, the discrepancy on i is at most $10\frac{m_i}{m^{1/3}}(\log n)^{1/3} + g_i < 20\frac{m_i}{m^{1/3}}(\log n)^{1/3} - g_i$. Hence, it follows from the way the algorithm deals with subsequent beads of color i, that the process will end with a balanced partition of the beads of each color i between the agents, allocating to each of them either $\lfloor m_i/2 \rfloor$ or $\lceil m_i/2 \rceil$ of these beads. As this argument works for every color, the algorithm produces a proper solution. Next, we prove the claims.

Proof of Claim 1:

Proof. Note that whenever some color i becomes critical, the term ψ_i that we drop from ψ is positive. Hence, it is enough to prove that $\psi(t) \geq \frac{\psi(t+1|\varepsilon_{t+1}=1)+\psi(t+1|\varepsilon_{t+1}=-1)}{2}$, where $\psi(t+1|\varepsilon_{t+1} = \chi)$ denotes the value of $\psi(t+1)$ if we choose $\varepsilon_{t+1} = \chi \in \{-1, 1\}$. It suffices to show that for every i, $\psi_i(t) \geq \frac{1}{2}[\psi_i(t+1|\varepsilon_{t+1} = 1] + \frac{1}{2}[\psi_i(t+1|\varepsilon_{t+1} = -1]$.

We proceed with the proof of this inequality. To do so, note that

$$\psi_i(t+1|\varepsilon_{t+1} = 1) = \frac{e^{\lambda(u_t + a_{t+1}/m_i)} + e^{-\lambda(u_t + a_{t+1}/m_i)}}{2} e^{\lambda^2 g(1 - s_t - a_{t+1}/m_i)/2},$$

and

$$\psi_i(t+1|\varepsilon_{t+1} = -1) = \frac{e^{\lambda(u_t - a_{t+1}/m_i)} + e^{-\lambda(u_t - a_{t+1}/m_i)}}{2} e^{\lambda^2 g(1 - s_t - a_{t+1}/m_i)/2}.$$

Therefore

$$\frac{\psi_i(t+1|\varepsilon_{t+1} = 1) + \psi_i(t+1|\varepsilon_{t+1} = -1)}{2} =$$

$$\frac{e^{\lambda u_t} + e^{-\lambda u_t}}{2} \cdot \frac{e^{\lambda a_{t+1}/m_i} + e^{-\lambda a_{t+1}/m_i}}{2} e^{\lambda^2 g(1 - s_t - a_{t+1}/m_i)/2}$$

$$\leq \frac{e^{\lambda u_t} + e^{-\lambda u_t}}{2} \cdot e^{\lambda^2 g(a_{t+1}/m_i)/2} e^{\lambda^2 g(1 - s_t - a_{t+1}/m_i)/2} = \frac{e^{\lambda u_t} + e^{-\lambda u_t}}{2} \cdot e^{\lambda^2 g(1-s_t)/2} = \psi_i(t),$$

as needed.

◀

Proof of Claim 2:

Proof. Let t be a cut made while color i is normal. To prove that the discrepancy on color i in absolute value is at most $10\frac{m_i}{m^{1/3}}(\log n)^{1/3}$, it suffices to prove $\psi_i(t) \leq \frac{1}{2}e^{\lambda \cdot 10(\frac{\log n}{m})^{1/3}} e^{\lambda^2 g(1-s_t)} = \frac{1}{2}e^{2\log n} e^{\lambda^2 g(1-s_t)}$. By Claim 1, $\psi(t) \leq \psi(0) = ne^{g\lambda^2/2}$. Hence, it is enough to prove that $ne^{g\lambda^2/2} \leq \frac{1}{2}e^{2\log n} e^{\lambda^2 g(1-s_t)}$. This is equivalent to $\lambda^2 g s_t/2 + \log 2 \leq 4\log n$. Since $s_t \leq 1$, we get $\lambda^2 g s_t/2 + \log 2 \leq \log n + \log 2 \leq 2\log n$, as needed.

◀

Lastly, we prove that the total number of cuts is $O(n(\log n)^{1/3} \cdot m^{2/3})$. The number of forced cuts cannot exceed $2n \cdot 20\frac{m_i}{m^{1/3}}(\log n)^{1/3} = O(m^{2/3}n(\log n)^{1/3})$. To bound the number of non-forced cuts, note that whenever we make such a cut, there is a color j such that the number of beads of this color on the interval created is exactly g_j. We call the cut j-tight for that respective color. It is easy to see that for every color i there are most $O(m^{2/3}(\log n)^{1/3})$ i-tight cuts. Hence, the total number of non-forced cuts is at most $O(m^{2/3}n(\log n)^{1/3})$. This completes the proof.

◀

4 Lower bounds

In this section we present the lower bounds for Necklace Halving in the online model.

4.1 A preliminary bound

We provide a $\Omega(\sqrt{m})$ lower bound for the number of cuts required in any online algorithm when the number of colors is $n = 2$ and there are m beads of each color. We need the following simple lemma, which is a special case of a more general elegant result of Tijdeman [24]. Since this special case is much simpler, we include its proof, for completeness.

▶ **Lemma 1.** *For every real $\gamma \in [0, 1]$ there is an infinite binary sequence $a_1, a_2, a_3, \ldots$ so that in every prefix of it $a_1, a_2, \ldots, a_j$ the number of elements a_i which are 1 deviates from γj by less than 1.*

Proof. By compactness it suffices to prove the existence of such a sequence of any finite length r. Consider the following system of linear inequalities in the variables $x_1, x_2, \ldots, x_r$: $0 \le x_i \le 1$ for all $1 \le i \le r$, and for every $j \le r$, $\lfloor \gamma j \rfloor \le \sum_{i=1}^{j} x_i \le \lceil \gamma j \rceil$. This system has a real solution $x_i = \gamma$ for every i and the matrix of coefficients of the constraints is totally unimodular. Hence there is an integral solution $x_i = a_i \in \{0, 1\}$ providing the required sequence. ◀

We use the following notation. During the algorithm let t denote the number of beads revealed so far. If a cut is made at this point, let x_t be the difference between the number of beads of color 1 allocated to agent 1 and the number of beads of color 1 allocated to agent 2. Define y_t similarly for beads of color 2. Let α_t, β_t denote the number of remaining beads of colors 1 and 2, respectively.

▶ **Lemma 2.** *Let Δ be a positive integer. Suppose that a cut is made at point t and $|x_t| = \Delta$ and assume that no bead of color 2 appeared so far. Then there exists an adversarial input that forces the Balancer to make at least $\Delta/4 = \Omega(\Delta)$ cuts.*

Proof. Without loss of generality assume that $x_t = \Delta > 0$. Note that by assumption $\beta_t = m$ and $\alpha_t < m$. Put $\gamma = \frac{m}{\alpha_t + m}$ and note that $\gamma > 1/2$. By Lemma 1 it is possible to choose an ordering of the remaining $\alpha_t + m$ beads of the necklace so that in every prefix of it of any length j, the number of beads of color 2 deviates from γj by less than 1. Since our online model allows the Balancer to see the next bead before the decision to make a cut preceding it we may have to change the first bead in this ordering, this still ensures that in any interval of length ℓ in the remainder of the necklace, the number of beads of color 2 deviates from $\gamma \ell$ by at most 2.

Suppose the Balancer cuts the remainder of the necklace and allocates the resulting intervals $R_1, \ldots, R_u$ to agent 1 and $T_1, \ldots, T_v$ to agent 2 to obtain a balanced allocation. For each one of these intervals I let $\ell(I)$ denote its length. By assumption at time t agent 1 has exactly Δ more beads than agent 2. Since at the end each agent has half of the beads (for simplicity we assume that m is even), $\sum_{i=1}^{v} \ell(T_i) - \sum_{j=1}^{u} \ell(R_j) = \Delta$.

By construction, the total number of beads of color 2 in all intervals T_i deviates from $\gamma \sum_{i=1}^{v} \ell(T_i)$ by at most $2v$. Similarly, the total number of beads of color 2 in all intervals R_j deviates from $\gamma \sum_{j=1}^{u} \ell(R_j)$ by at most $2u$. As these two numbers should be equal it follows that

$$\gamma\Delta = \gamma(\sum_{i=1}^{v} \ell(T_i) - \sum_{j=1}^{u} \ell(R_j)) \leq 2u + 2v$$

This implies that $2(u+v) \geq \gamma\Delta > \Delta/2$ and as the number of cuts is at least $u+v$ the desired result follows.

◀

The last lemma easily implies the following.

▶ **Theorem 3.** *There exists an adversarial input that forces any deterministic algorithm for Online Necklace Halving with $n = 2$ colors to make $\Omega(\sqrt{m})$ cuts in order to obtain a proper solution.*

Proof. Put $\Delta = \sqrt{m}$ and proceed by revealing only beads of color 1. By Lemma 2, if after a cut at some t, $|x_t| > \sqrt{m}$, the desired result follows. Otherwise it is clear the number of beads between any two consecutive cuts is less than $2\sqrt{m}$, implying that the total number of cuts made by the Balancer is $\Omega(\sqrt{m})$.

◀

4.2 A nearly tight bound

▶ **Theorem 4.** *An adversary can force any deterministic algorithm for Online Necklace Halving with $n = 3$ colors and m beads of each color to make $\Omega(m^{2/3})$ cuts.*

Proof. As in the previous subsection, let x_t denote the discrepancy between the number of beads of color 1 allocated to agent 1 and that allocated to agent 2 after cut t, and let y_t denote the corresponding discrepancy for color 2, where color 3 will be kept as a potential threat. We proceed by revealing only beads of the first two colors. By Lemma 2 with $\Delta = m^{2/3}$ the Balancer needs to maintain $|x_t|, |y_t| \leq m^{2/3}$, since otherwise the adversary can force $\Omega(m^{2/3})$ cuts, using beads of the third color. Hence, we assume that during the process of revealing the initial $m + 4m^{2/3}$ beads of the necklace x_t, y_t stay in the above range after each cut.

Define a potential function

$$M(x,y) = x^2 + y^2 + 5m^{2/3}(x-y)$$

After a cut with $v_t = (x_t, y_t) = (x, y)$ define $\gamma = \frac{10m^{2/3}-4y}{20m^{2/3}+4(x-y)}$. Note that $0 < \gamma < 1$, as $|x|, |y| \leq m^{2/3}$. By Lemma 1 it is possible to order the remaining part of the first $m + 4m^{2/3}$ beads of the necklace so that in each prefix of any length j of this remaining part the number of beads of color 1 deviates from γj by less than 1 and the number of beads of color 2 deviates by less than 1 from $(1-\gamma)j$. As the first bead of this remaining part has been observed already by the Balancer we may need to change one bead in this ordering, getting a deviation of less than 2 in each prefix. This means that if the next cut will be made after some j

additional beads, the vector $p = (p_1, p_2)$ of additional beads of colors 1 and 2, respectively, can be written as a sum of the vector $p' = (\gamma j, (1-\gamma)j)$ and an error vector $\delta = (\delta_1, \delta_2)$ of ℓ_∞-norm smaller than 2. We get that

$$M(v_t + p') - M(v) = p_1'^2 + p_2'^2 + 2xp_1' + 2yp_2' + 5m^{2/3}p_1' - 5m^{2/3}p_2' =$$

$$= p_1'^2 + p_2'^2 + \frac{1}{2}[p_1' \cdot (10m^{2/3} + 4x) - p_2' \cdot (10m^{2/3} - 4y)] = p_1'^2 + p_2'^2 \geq \frac{1}{2}j^2$$

and similarly,

$$M(v - p') - M(v) = p_1'^2 + p_2'^2 + \frac{1}{2}[-p_1' \cdot (10m^{2/3} + 4x) + p_2' \cdot (10m^{2/3} - 4y)] = p_1'^2 + p_2'^2 \geq \frac{1}{2}j^2$$

A simple computation using the fact that $|x|, |y| \leq m^{2/3}$ and that a similar bound holds after adding or subtracting the vector p' shows that adding or subtracting the vector δ can decrease the value of M by less than $15m^{2/3}$. Therefore, we get

$$M(v_t \pm p) - M(v_t) \geq j^2/2 - 15m^{2/3}$$

which implies $M(v_{t+1}) - M(v_t) \geq j^2/2 - 15m^{2/3}$, with a cut of j beads.

Suppose that we have r cuts among the first $m + 4m^{2/3}$ beads of the necklace, and the lengths of the resulting intervals are $j_1, j_2, \ldots, j_r$. Since throughout the process $|x_t|, |y_t| \leq m^{2/3}$, it follows that $M(x_t, y_t) \leq 12m^{2/3}$. On the other hand by the above discussion the value of M at the end is at least $\sum_{i=1}^r \frac{j_i^2}{2} - 15m^{2/3}r$. Since $\sum_{i=1}^r j_i \geq m$ (as we cannot have $4m^{2/3}$ consecutive beads with no cut among them), it follows, by Cauchy-Schwartz, that $\sum j_i^2 \geq \frac{m^2}{r}$. This implies that

$$\frac{1}{2}\frac{m^2}{r} - 15rm^{2/3} \leq 12m^{4/3}$$

showing that $r = \Omega(m^{2/3})$, as needed.

◀

Remark: For $n > 3$ colors with m beads of each color one can consider a necklace consisting of $\lfloor n/3 \rfloor$ segments with at least 3 colors in each of them. The above argument shows that it is possible to force $\Omega(m^{2/3})$ cuts in each segment, implying an $\Omega(nm^{2/3})$ lower bound. Thus, for n colors, the gap between our lower and upper bounds for the number of cuts required is only a factor of $\Theta((\log n)^{1/3})$.

5 Extensions and open problems

We conclude with some generalizations of the algorithms presented and the lower bounds obtained, and with comments on some of the questions that remain open.

5.1 Generalizations

In this section, we present our online and offline results for the general case of k agents.

▶ **Theorem 5.** *There exists an efficient, deterministic, offline algorithm that provides a proper solution to the Necklace Splitting problem, making at most* $n(k-1)\lceil 4 + \log_2(3km) \rceil$ *cuts.*

Proof. As in the proof of Theorem 1, we first convert the Necklace Splitting instance into a continuous instance J, and obtain a solution with absolute discrepancy at most $\frac{\varepsilon}{2k} = \frac{1}{2km}$, possibly making some floating cuts. Then, to obtain a proper solution for the discrete instance, we shift the floating cuts by solving a network flow problem.

To obtain the solution to the continuous instance J, we recursively apply a modified version of the algorithm that makes cuts on the continuous necklace from Theorem 1. Define $\varepsilon' = \varepsilon/3k = \frac{1}{3km}$, and divide the k players into two disjoint groups A, B, with $\lfloor k/2 \rfloor$ agents and $\lceil k/2 \rceil$ agents respectively. Think of A, B as two agents and split the continuous necklace among them. By following the algorithm in the proof of Theorem 1, one can make $\leq n(2 + \lceil \log_2 \frac{1}{\varepsilon'} \rceil)$ cuts and split the interval so that A gets $\frac{\lfloor k/2 \rfloor}{k} \pm \varepsilon'/2$ of each measure i. We can do so by starting with all floating coefficients equal to $\frac{\lfloor k/2 \rfloor}{k}$ instead of $\frac{1}{2}$ and by following the proof of Theorem 1. Repeat the same procedure for the groups A and B recursively, splitting the share of A among its $|A|$ members and doing the same for B. In the end, the error can be bounded by $\varepsilon' + \frac{2}{3}\varepsilon' + \frac{2}{3} \cdot \frac{4}{7}\varepsilon' + ... < 3\varepsilon'$. If we denote by $T(k)$ the number of cuts made to obtain absolute discrepancy $\leq \varepsilon'$ for a continuous instance with n types and k agents, then $T(2) = n\lceil \log_2 \frac{1}{\varepsilon'} + 2 \rceil$, and $T(k) = T(\lfloor k/2 \rfloor) + T(\lceil k/2 \rceil) + n\lceil \log_2 \frac{1}{\varepsilon'} + 2 \rceil$, which gives that the number of cuts made for this split is $T(k) = n(k-1)\lceil 2 + \log_2(3km) \rceil)$.

Hence, we have obtained a proper solution for the continuous instance J, making $n(k - 1)\lceil 2 + \log_2(3km) \rceil)$ cuts, yet we have to handle floating cuts. We categorize each floating cut by the color of the interval in whose interior it lies. For each color i, we handle the corresponding floating cuts. First, note that if $k > m_i$, we can shift each floating cut on color i to one of the ends of the i-interval in such a way that no agent gets more than one bead of color i and this will provide discrepancy at most 1 on color i without creating any additional cuts. Hence, we may assume $m_i \geq k$.

We use a network flow algorithm to decide, for each bead of color i that does not fully belong to one agent, to whom it should be allocated. Define a directed graph G_i, with vertices s, the source, t, the sink, V_i, representing the set of beads of color i, and H, the set of vertices representing the agents. Let E be the set of edges with

$$E = \{(s, v), v \in V_i\} \cup \{(h, t), h \in H\} \cup \{(v, h), \text{agent h owns a share of bead v}\}$$

All edges $\{(s, v), v \in V_i\}$ have capacity 1 and lower bound 1. Each edge (v, h) has capacity 1 and lower bound 0. Finally, for each edge (h, t), set the capacity to be $\lceil x_h \rceil$ and the lower bound to be $\lfloor x_h \rfloor$, where x_h is the quantity of type i allocated to agent h in the solution to the continuous instance. Now, if we assign each edge (v, h) a value equal to the share of bead v allocated to agent h in the continuous solution and each edge (h, t) the value x_h, this is a legal flow. Hence, there exists an integral legal flow in the network, and it is well known that one can find such a flow efficiently. Note that an integral flow corresponds to a distribution of the beads of color i where no additional cut is made and the absolute discrepancy is at most 1 if $k \nmid a_i$ and at most 2 if $k | a_i$. Thus, the integral flow determines which agent gets each of the contested beads of color i, corresponding to a shift of each floating cut to one of the ends of the bead it crosses.

If $k|a_i$, the continuous solution could give some agent a a share of $x_a = a_i/k - \varepsilon_1$ and some agent b a share $x_b = a_i/k + \varepsilon_2$, for small positive values $\varepsilon_1, \varepsilon_2$. In this case, the integral network flow solution could give agent a $a_i/k - 1$ beads and agent b $a_i/k + 1$ beads of color i. As a_i/k is an integer, the number of agents receiving $a_i/k + 1$ beads is the same as the number of agents receiving $a_i/k - 1$. Hence, we can make at most $2k$ cuts after the shift is done to obtain discrepancy 0. We perform the shifting procedure for every color i, and obtain a proper solution with at most $nk + n(k-1)\lceil 2 + \log_2(3km) \rceil < n(k-1)\lceil 4 + \log_2(3km) \rceil$, as needed. This completes the proof. The network flow argument follows the approach in [5]. ◀

▶ **Theorem 6.** *There exists an efficient, deterministic, online algorithm that provides a proper solution for the Necklace Splitting problem, making at most* $\tilde{O}(nk^{1/3} \cdot m^{2/3})$ *cuts.*

Proof. Note that the result is trivial for $k > m$. For $k \le m$, we again use the idea of defining a potential function ϕ and a function ψ that is an upper bound for ϕ and is computable efficiently. Instead of having one pair of functions ϕ_i, ψ_i for each color i, we now have $\binom{k}{2}$ such functions, one for each pair of agents. For each color i and agents $p \neq q$, define $\phi_i^{p,q} = \mathbb{E}\left[\frac{e^{\lambda X_{p,q,i}/m_i} + e^{-\lambda X_{p,q,i}/m_i}}{2}\right]$, where $X_{p,q,i}$ is the random variable of the difference between the number of beads of color i given to agent p and that of agent q. The relevant random distribution here assigns every newly created interval to one of the k agents with equal probability which is $1/k$. The quantity $g = g(n,k,m)$ is defined here as $g = \frac{1}{m^{2/3}k(\log(nk))^{1/3}}$, and each g_i, the maximum number of beads of color i allowed between two consecutive cuts as $g_i = m_i g$. We say color i is *critical* when the number of remaining beads of this color is at most $20k^{1/3}m^{2/3}$.

The function $\psi_i^{p,q}$ is defined by

$$\psi_i^{p,q}(t) = \frac{e^{\lambda x_{t,i}^{p,q}} + e^{-\lambda x_{t,i}^{p,q}}}{2} \cdot e^{2\lambda^2 g(1-s_t)/k}$$

where s_t is, as before, the proportion of beads of color i allocated already, and $x_{t,i}^{p,q}$ is the discrepancy between p and q on color i after cut t divided by m_i.

The main difference required here is the replacement of the inequality $\cosh(\lambda a) \le e^{\lambda^2 a^2/2}$ by the following inequality which holds whenever, say, $\lambda a \le 1$:

$$\frac{k-2}{k}e^{\lambda \cdot 0} + \frac{1}{k}e^{\lambda a} + \frac{1}{k}e^{-\lambda a} = 1 + \frac{2}{k}(\cosh(\lambda a) - 1)$$

$$\le 1 + \frac{2}{k}(e^{\lambda^2 a^2/2} - 1) \le 1 + \frac{2}{k}\frac{2\lambda^2 a^2}{2} = 1 + \frac{2\lambda^2 a^2}{k} \le e^{2\lambda^2 a^2/k}.$$

Each $\phi_i^{p,q}$ is bounded using the fact that each of the intervals created has at most $g_i = m_i g$ beads of color i for every i. By the inequality applied with $a \le g$ and $\lambda = \frac{(k/m)^{1/3}}{4g}$ (ensuring that indeed $\lambda a \le \frac{(k/m)^{1/3}}{4} < 1/2$), it follows that if every interval generated is allocated to an agent in order to minimize $\psi = \sum_{p,q \in [k],\ p \neq q,\ i \in [n]} \psi_i^{p,q}$, then the function ψ never increases during the algorithm. As

$$\psi(0) < nk^2 e^{2\lambda^2 g/k} = nk^2 e^{\varepsilon^2/8gk} < \frac{e^{\lambda\varepsilon/k}}{2}$$

the computation shows that at the end the absolute discrepancy is $\le \varepsilon/k$. We omit the details. ◀

Next, we present two simple special cases where we obtain proper solutions efficiently with the optimal number of cuts, $n(k-1)$. In the first case, the number of beads of each color is equal to k, the number of agents. In the second case, we set the number of colors to be $n = 2$.

▶ **Proposition 1.** *There exists an efficient algorithm that solves any instance of Necklace Splitting for n colors and k agents where there are exactly k beads of each color, making at most $n(k-1)$ cuts.*

Proof. Traverse the necklace once bead by bead and cut between any pair of consecutive beads unless the second one is the first appearance of a bead of color i for some $i \in [n]$. After each cut made, if S is the set of colors present in the newly created interval J, we allocate J to an agent that has not received up to that point any beads of any color in S. To show that after each cut such an agent exists, first note that by the description above, no agent receives two beads of the same color. If J contains only one bead and its color is i, there must exist an agent who has not received any bead of color i up to that point, as there are as many agents as beads of color i. If J has $p \geq 2$ beads, of colors $c_1, ..., c_p \in [n]$ appearing in this order, we can still give it to an agent that has not received any bead of color c_1, since each of the other beads in J has a color that has not appeared before.

It thus follows that with this allocation rule each agent gets exactly 1 bead of each color. To prove the upper bound on the number of cuts, note that for each $i \in [n]$, we never cut right before the first bead of color i that appears on the necklace. Hence, there are exactly $n-1$ beads (besides the very first one) with no cut right before them. Since there are $kn-1$ points between consecutive beads the algorithm makes exactly $kn-1-(n-1) = n(k-1)$ cuts. ◀

▶ **Proposition 2.** *There exists an efficient algorithm that solves any instance of Necklace Splitting for $n = 2$ colors and k agents, making at most $2(k-1)$ cuts.*

Proof. We first consider the case when k divides both m_1, m_2, where m_i is the number of beads of color i. Given a necklace with m_1 beads of color 1 and m_2 beads of color 2 consider it as a circular necklace. By the discrete intermediate value theorem there is a circular arc of $(m_1+m_2)/k$ beads containing exactly m_1/k beads of color 1 (and hence also exactly m_2/k beads of color 2). Cut in the ends of this circular arc, assign it to the first agent, and continue inductively. Clearly, every agent gets the same number of beads of each color.

To extend the proof for general m_1, m_2, write $m_1 = kp + r$ and $m_2 = kq + s$. We look for a circular arc of $\lceil \frac{m_1}{k} \rceil + \lceil \frac{m_2}{k} \rceil$ beads containing exactly $\lceil \frac{m_1}{k} \rceil$ beads of color 1 (and hence also exactly $\lceil \frac{m_2}{k} \rceil$ beads of color 2). If $r \neq 0$, the agent to whom we distribute the arc gets $p+1$ beads of color 1. Similarly, if $s \neq 0$, the agent gets $q+1$ beads of color 2. Hence, by inductively finding a suitable arc and cutting it from the necklace, at the end of the process, the first r agents will get $p+1$ beads of color 1 and the rest p. Similarly, the first s agents will get $q+1$ beads of color 2 and the rest q. ◀

5.2 Connections to ε-Consensus Splitting

Our results easily extend to the ε-Consensus Splitting problem with non-atomic probability measures whose density functions are piecewise linear. This is stated in the next two theorems whose detailed proofs are provided in the full version, [3].

▶ **Theorem 7.** *There exists an efficient, deterministic, offline algorithm that provides a solution to the ε-Consensus Splitting problem, making at most $n(k-1)\lceil 4 + \log_2(3km) \rceil$ cuts, provided that the density functions of the probability measures are piecewise linear.*

▶ **Theorem 8.** *There exists an efficient, deterministic, online algorithm that provides a solution for the ε-Consensus Splitting problem, making at most $O(\frac{kn\log(nk)}{\varepsilon^2})$ cuts, provided that the density functions of the probability measures are piecewise linear.*

Note that for $k = 2$ agents, the number of cuts resulting from the algorithm corresponding to Theorem 8 is $O(\frac{n \log n}{\varepsilon^2})$. The proof of Theorem 2 relies on using this algorithm for $k = 2$ agents with $\varepsilon = \Theta((\frac{\log n}{m})^{1/3})$.

5.3 Open questions

Theorem 1 provides a proper solution to the offline version for $k = 2$ agents by making a number of cuts that depends logarithmically on m, the maximum number of beads of a color. It would be interesting to see if this dependency can be improved asymptotically.

Another open question arises in the context of the Online Necklace Halving problem for $n = 2$ colors, where the lower bound for the number of cuts is only $\Omega(\sqrt{m})$, whereas the upper bound for the number of cuts produced by our algorithm is $O(m^{2/3})$. Lastly, for the general case of n colors for the online version of Necklace Halving there is a $\Theta((\log n)^{1/3})$ gap between the lower bound and the algorithm we provided. It will be interesting to close these gaps.

References

1 Noga Alon. Splitting necklaces. *Advances in Mathematics*, 63(3):247–253, 1987.

2 Noga Alon. Non-constructive proofs in Combinatorics. *Proceedings of the International Congress of Mathematicians (ICM)*, 63:1421–1429, 1990.

3 Noga Alon and Andrei Graur. Efficient Splitting of Measures and Necklaces. *arXiv:2006.16613*, 2020.

4 Noga Alon, Michael Krivelevich, Joel H. Spencer, and Tibor Szabó. Discrepancy Games. *The Electronic Journal of Combinatorics*, 12(1), 2005.

5 Noga Alon, Dana Moshkovitz, and Muli Safra. Algorithmic construction of sets for k-restrictions. *ACM Transactions on Algorithms*, 2:153–177, 2006.

6 Noga Alon and Douglas B West. The Borsuk-Ulam Theorem and Bisection of Necklaces. *Proceedings of the American Mathematical Society*, 98(4):623–628, 1986.

7 Nikhil Bansal. Constructive Algorithms for Discrepancy Minimization. *Proc. 51st Symposium on Foundations of Computer Science (IEEE)*, pages 3–10, 2010.

8 Nikhil Bansal and Joel H. Spencer. Deterministic Discrepancy Minimization. *Algorithmica*, 67:451–471, 2013.

9 Nikhil Bansal and Joel H. Spencer. On-line Balancing of Random Inputs. *Random Structures and Algorithms*, 57(4):879–891, 2020.

10 Sandeep N. Bhatt and Frank T. Leighton. A Framework For Solving VLSI Graph Layout Problems. *Journal of Computer and System Sciences*, 28(2):300–343, 1984.

11 Sandeep N. Bhatt and Charles E. Leiserson. How to assemble tree machines. *Proceedings of the 14th Symposium on the Theory of Computing, San Francisco*, pages 99–104, 1981.

12 Paul Simon Bonsma, Thomas Epping, and Winfried Hochstättler. Complexity results on restricted instances of a paint shop problem for words. *Discrete Appl. Math*, 154(9):1335–1343, 2006.

13 Steven J. Brams and Alan D. Taylor. Fair division: From cake-cutting to dispute resolution. *Cambridge University Press*, 1996.

14 Bruno Codenotti, Amin Saberi, Kasturi Varadarajan, and Yinyu Ye. The complexity of equilibria: Hardness results for economies via a correspondence with games. *Theoretical Computer Science*, 408:188–198, 2008.

15 Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. The Complexity of Computing a Nash Equilibrium. *Theoretical Computer Science*, 39(1):195–259, 2009.

16 Aris Filos-Ratsikas, Soren Kristoffer Stiil Frederiksen, Paul W. Goldberg, and Jie Zhang. Hardness Results for Consensus Halving. *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 24:1–24:16, 2018.

17 Aris Filos-Ratsikas and Paul W. Goldberg. Consensus Halving is PPA-Complete. *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC)*, pages 51–64, 2018.

18 Aris Filos-Ratsikas and Paul W. Goldberg. The Complexity of Splitting Necklaces and Bisecting Ham Sandwiches. *Proceedings of the 51st Annual ACM Symposium on Theory of Computing (STOC)*, pages 638–649, 2019.

19 Aris Filos-Ratsikas, Alexandros Hollender, Katerina Sotiraki, and Manolis Zampetakis. Consensus Halving: Does it Ever Get Easier? *arXiv:2002.11437*, 2020.

20 Charles H. Goldberg and Douglas B. West. Bisection of circle colorings. *SIAM J. Algebraic Discrete Methods*, 6:93–106, 1985.

21 Charles R. Hobby and John R. Rice. A moment problem in L_1 approximation. *Proceedings of the American Mathematical Society*, 16(4):665–670, 1965.

22 Frédéric Meunier. Simplotopal maps and necklace splitting. *Discrete Mathematics*, 323:14–26, 2014.

23 Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48:498–532, 1994.

24 Robert Tijdeman. On a distribution problem in finite and countable sets. journal of combinatorial theory. *Journal of Combinatorial Theory, Series A*, 15(2):129–137, 1973.