On the complexity of cake cutting

Gerhard J. Woeginger\textsuperscript{a,*}, Jiří Sgall\textsuperscript{b}

\textsuperscript{a}Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
\textsuperscript{b}Mathematical Institute of the Academy of Sciences of the Czech Republic, Žitná 25, CZ-11567 Praha 1, Czech Republic

Received 29 June 2005; received in revised form 1 July 2006; accepted 5 July 2006
Available online 19 December 2006

Abstract

In the cake cutting problem, \( n \geq 2 \) players want to cut a cake into \( n \) pieces so that every player gets a ‘fair’ share of the cake by his/her own measure.

We prove that in a certain, natural cake cutting model, every fair cake division protocol for \( n \) players must use \( \Omega(n \log n) \) cuts and evaluation queries in the worst case. Up to a small constant factor, this lower bound matches a corresponding upper bound in the same model obtained by Even and Paz, from 1984.

\( \copyright \) 2007 Published by Elsevier B.V.

Keywords: Fair division; Cake cutting

1. Introduction

1.1. Problem description

In the cake cutting problem, there are \( n \) players \( 1, \ldots, n \) and a certain resource \( C \) (hereinafter referred to as the cake). In agreement with much of the cake cutting literature, we will assume throughout the paper that \( C = [0, 1] \) is the unit interval and that the cuts divide the cake into subintervals. Every player \( p \) (\( 1 \leq p \leq n \)) has his/her own private measure \( \mu_p \) on sufficiently many subsets of \( C \). These measures \( \mu_p \) are assumed to be well behaved; this means that they are:

- Defined: on all finite unions of intervals.
- Non-negative: For all \( X \subseteq C \), \( \mu_p(X) \geq 0 \).
- Additive: For all disjoint subsets \( X, X' \subseteq C \), \( \mu_p(X \cup X') = \mu_p(X) + \mu_p(X') \).
- Divisible: For every measurable \( X \subseteq C \) and \( 0 \leq \lambda \leq 1 \), there exists \( X' \subseteq X \) with \( \mu_p(X') = \lambda \mu_p(X) \).
- Normalized: \( \mu_p(C) = 1 \).

All these assumptions are standard assumptions in the cake cutting literature, sometimes subsumed in a concise statement that each \( \mu_p \) is a probability measure defined on Lebesgue measurable sets and absolutely continuous

\* Corresponding author.

E-mail addresses: gwoegi@win.tue.nl (G.J. Woeginger), sgall@math.cas.cz (J. Sgall).

1572-5286/$ - see front matter \( \copyright \) 2007 Published by Elsevier B.V.
doi:10.1016/j.disopt.2006.07.003
with respect to Lebesgue measure. We stress that the divisibility of $\mu_p$ forbids concentration of the measure in one or more isolated points. As one consequence of this, corresponding open and closed intervals have the same measure, and thus we do not need to be overly formal about the endpoints of intervals.

The cake $C$ is to be divided among the $n$ players according to some fixed cake division protocol, which is an interactive procedure for the players that guides and controls the process of division of the cake $C$. Typically it consists of cut requests like “Cut cake piece $Z_1$ into two equal pieces, according to your measure!” and evaluation queries like “Is your measure of cake piece $Z_1$ less than, greater than, or equal to your measure of cake piece $Z_2$?” or “What is your measure of cake piece $Z_1$?”. Eventually, the protocol assigns to the players pairwise disjoint subsets of the cake. A cake division protocol is not a priori aware of the measures $\mu_p$, but it will learn something about them during its execution, and the answers may influence which cut requests and evaluation queries are issued in the future (that is, the protocol is adaptive). A strategy of a player is an adaptive sequence of moves consistent with a given protocol.

A cake division protocol is (perfectly) fair if every player $p$ has a strategy that guarantees him/her a piece of size at least $\mu_p(C)/n$ according to his/her own measure $\mu_p$. So, if $n=1$ players were to plot with coordinated moves against a single player, then this single player will still be able to get his/her share of $\mu_p(C)/n$. This is called simple fair division in the literature. The complexity of a cake cutting protocol is generally measured by the number of cuts performed in the worst case. We discuss the exact definition of this measure and the notion of a cake cutting protocol later.

The central open problem in this area is whether there exist perfectly fair $n$-player protocols that only use $O(n)$ cuts. This problem was explicitly formulated by Even and Paz [2], and essentially goes back to Steinhaus [8]. The general belief is that no such protocol exists.

1.2. Previous results

In the 1940s, the Polish mathematicians Banach and Knaster designed a simple fair cake division protocol that uses $O(n^2)$ cuts in the worst case; this protocol was explained and discussed in 1948 by Steinhaus [8]. In 1984, Even and Paz [2] used a divide-and-conquer approach to construct a better deterministic protocol that only uses $O(n \log n)$ cuts in the worst case. Remarkably, Even and Paz [2] also design a randomized protocol that uses an expected number of $O(n)$ cuts. Tighter results are known for small values of $n$: For $n=2$ players, the Steinhaus protocol yields a perfectly fair protocol with a single cut. For $n=3$ and $n=4$ players, Even and Paz [2] present perfectly fair protocols that make at most three and four cuts, respectively. Webb [10] presents a perfectly fair protocol for $n=5$ players with six cuts.

The problem of establishing lower bounds for cake cutting goes back at least to Banach (see [8]). Even and Paz [2] explicitly conjecture that there does not exist a fair deterministic protocol with $O(n)$ cuts. Robertson and Webb [7] support and strengthen this conjecture by saying they “would place their money against finding a substantial improvement on the $n \log n$ [upper] bound”.

One basic difficulty in proving lower bounds for cake cutting is that most papers derive upper bound results and to do that, they simply describe a certain procedure that performs certain steps, and then establish certain nice properties for it, but they do not provide a formal definition or a framework. Even and Paz [2] give a proof that for $n \geq 3$, no protocol with $n-1$ cuts exists; since $n-1$ cuts are the smallest possible number, such protocols would need to be rather special (in particular they would have to assign a single subinterval to each player) and not much formalism is needed. Webb [10] shows that for $n=5$, no perfectly fair protocol exists that uses only five cuts, and thus his protocol with six cuts is optimal. Only recently, Robertson and Webb [6,7] give a more precise definition of a protocol that covers all the protocols given in the literature. This definition avoids some pathological protocols, but it is still quite general and no superlinear lower bounds are known.

A recent paper [4] by Magdon-Ismail, Busch and Krishnamoorthy proves an $\Omega(n \log n)$ lower bound for a certain non-standard cake cutting model: The lower bound does not hold for the number of cuts performed or evaluation queries, but for the number of comparisons needed to administer these cuts. For more information on this fair cake cutting protocol and on many of its variants, we refer the reader to the books by Brams and Taylor [1] and by Robertson and Webb [7].

1.3. Our contribution and organization of the paper

In Section 2 we first formally define the Robertson–Webb cake cutting model, and then we introduce a certain restriction of this model. The restriction has two requirements: (i) each player receives a single subinterval of the
cake, and (ii) the evaluation queries are counted towards the complexity of the protocol together with cuts. Our model is also general enough to cover the \(O(n \log n)\) cut deterministic protocol of Even and Paz [2], and we believe that it is fairly natural. We discuss some of the issues related to the formal definition of both models, and we put it into context with other results from the cake cutting literature.

In Section 3 we then show that in our restricted model, every deterministic fair cake division protocol for \(n\) players must use \(\Omega(n \log n)\) cuts and evaluation queries in the worst case. This result yields the first superlinear lower bound on the number of cuts for simple fair division (in our restricted model), and it also provides a matching lower bound for the result in [2]. Section 4 gives a discussion and some open problems.

2. The restricted cake cutting model

A general assumption in the cake cutting literature is that at the beginning of an execution a protocol has absolutely no knowledge about the measures \(\mu_p\), except that they are defined on intervals, non-negative, additive, divisible, and normalized. The protocol issues queries to the players, the players react, the protocol observes their reactions, issues more queries, observes more reactions, and so on, and so on, and in the end the protocol assigns the cake pieces to the players.

2.1. Definition of the Robertson–Webb model

We remind the reader that the cake \(C\) is represented by the unit interval. For a real number \(\alpha\) with \(0 \leq \alpha \leq 1\), the \(\alpha\)-point of a player \(p\) is the infimum of all numbers \(x\) for which \(\mu_p([0, x]) = \alpha\) and \(\mu_p([x, 1]) = 1 - \alpha\). In the Robertson–Webb model, the following two query types are allowed.

- **Cut**\((p; \alpha)\): Player \(p\) cuts the cake at his/her \(\alpha\)-point (where \(0 \leq \alpha \leq 1\)). The value \(x\) of the \(\alpha\)-point is returned to the protocol.
- **Eval**\((p; x)\): Player \(p\) evaluates the value of the cut \(x\), where \(x\) is one of the cuts previously performed by the protocol. The value \(\mu_p([0, x])\) is returned to the protocol.

The protocol can also assign an interval to a player; by doing this several times, a player may end up with a finite union of intervals.

- **Assign**\((p; x_i, x_j)\): Player \(p\) is assigned the interval \([x_i, x_j]\), where \(x_i \leq x_j\) are two cuts previously performed by the protocol or 0 or 1. All the intervals assigned to the players are required to be pairwise disjoint.

The complexity of a protocol is given by the number of cuts performed in the worst case, that is, all evaluation queries may be issued for free.

2.2. Definition of our restricted model

In our restricted model, the additional two restrictions are the following: First, **Assign**\((p; x_i, x_j)\) is used only once for each \(p\). Hence, in the restricted model every player ends up with a single (contiguous) subinterval of the cake. Secondly, the complexity of a protocol is given by the number of cuts plus the number of evaluation queries. In other words, each evaluation query contributes to the complexity the same as a cut. Note that this also covers counting only the number of cuts in protocols that do not use evaluation queries at all.

2.3. Discussion of our restricted model

The currently best deterministic protocol for fair division of Even and Paz [2] does not need evaluation queries and assigns single intervals; we will provide an almost matching bound within these restrictions. Nevertheless, both restrictions of our model are essential for our lower bound construction. A number of protocols in the literature (see [5,6]) do use evaluation queries, sometimes even a quadratic number of them. The randomized protocol of Even and Paz [2] also uses evaluation queries in addition to an expected number of \(O(n)\) cuts. The expected number of evaluation queries in this protocol is \(\Theta(n \log n)\).
We feel that the other restriction, that every player must receive a single, contiguous subinterval of the cake, is perhaps even stronger. By imposing this restriction, it seems that we severely cut down the set of possible protocols; in particular, for some instances, the solution is essentially unique (see our lower bound). Note, however, that with a small number of exceptions [3,11,12] most of the known fair discrete cake cutting protocols from the literature produce solutions where every player ends up with a contiguous subinterval. For instance, all the protocols in [2,5,6,8–10] have this property. In particular, the divide-and-conquer protocols of Even and Paz [2], both deterministic and randomized, assign a single contiguous subinterval to each player, as noted above. (Note: We stress that the picture changes completely when one moves on to so-called envy-free cake cutting protocols. Most of the known envy-free protocols do not assign contiguous subintervals to the players. However, being envy-free will not be considered in this paper.)

2.4. Discussion of the Robertson–Webb model

The Robertson–Webb model restricts the format of queries to cuts at \( \alpha \)-points and evaluation queries. This restriction is severe, but it is crucial and essentially unavoidable. Such a restriction must be imposed in one form or the other, just to prevent certain uninteresting ‘cheating’ protocol types from showing up with a linear number of cuts. Consider the following ‘cheating’ protocol:

**Phase 1.** Every player makes a cut that encodes his/her \( i/n \)-points with \( 1 \leq i \leq n - 1 \) (just fix any bijective encoding of \( n - 1 \) real numbers from [0, 1] into a single number from [0, 1]).

**Phase 2.** The protocol executes the Banach–Knaster protocol in the background (the Banach–Knaster protocol [8] is a fair protocol that only needs to know the positions of the \( i/n \)-points). That is, the protocol determines the relevant cuts without performing them.

**Phase 3.** The protocol tells the players to perform the relevant \( n - 1 \) cuts for the Banach–Knaster solution. If a player does not perform the cut that he/she announced during the first phase, he/she is punished and receives an empty piece (and his/her piece is added to the piece of some other player).

Clearly, every honest player will receive a piece of size at least \( 1/n \). Clearly, the protocol also works in the friendly environment where every player truthfully executes the orders of the protocol. And clearly, the protocol uses only \( 2n - 1 \) cuts — a linear number of cuts. Moreover, in this protocol every player ends up with a single subinterval of the cake. In cake cutting models that allow announcements of arbitrary real numbers, the cuts in Phase 1 can be replaced by direct announcements of the \( i/n \)-point positions; this yields fair protocols with only \( n - 1 \) cuts. These ‘cheating’ protocols are artificial, unnatural, and uninteresting, and it is hard to accept them as valid protocols. In the Robertson–Webb model they cannot occur, since they violate the form of queries. (One could try to argue that the players might disobey the queries and announce any real number. However, this fails, since the definition of a protocol entails only that a player that honestly answers allowed queries should get a fair share.)

Another important issue is that in the Robertson–Webb model it is sufficient to assume that all players are honest, that is, that they execute the commands “Cut at an \( \alpha \)-point” and evaluation queries truthfully. Under this assumption all of them get a fair share. Often in the literature, a protocol has no means of enforcing a truthful implementation of these cuts by the players, since the players may cheat, and lie, and try to manipulate the protocol; the requirement is then that any honest player get a fair share, regardless of the actions of the other players. In the Robertson–Webb model, any protocol that works for honest players can be easily modified to the general case as follows. As long as the answers of a player are consistent with some measure, the protocol works with no change, as it assigns a fair share according to this measure (and if the player has a different measure, he/she lied and has no right to complain). If an inconsistency is revealed (for instance, a violation of non-negativity), the protocol has to be modified to ignore the answers from this player (or rather replace them by some trivial consistent choices).

Of course, in general, the honesty of players is not a restriction on the protocol, but a restriction on the environment. Thus it is of no concern for our lower bound argument which uses only honest players. On the other hand, it simplifies the presentation of protocols.

In some details our description of the model is different than that of Robertson and Webb. First, in place of evaluation queries they assume that after a cut is made, its value in all the players’ measures becomes known. This covers all the possible evaluation queries, so it is clearly equivalent if we do not count the number of these queries. However, the number of evaluations is an interesting parameter, and this is why we chose our formalism.
Secondly, Robertson and Webb also allow cut requests of the form “cut this piece into two pieces with a given ratio of their measures”. This is very useful for an easy formulation of recursive divide-and-conquer protocols. Again, once free evaluation queries are allowed, this is no longer general, as we know all the measures of all the existing pieces. Even if we count evaluation queries, we can first evaluate the cuts that created the piece, and then such a non-standard cut is replaced by two evaluations and a standard cut at some α-point.

Thirdly, instead cutting at the α-point, Robertson and Webb allow an honest player to return any x with \( \mu_p([0, x]) = \alpha \); we require the answer which is the minimum of the honest answers according to Robertson and Webb. This is a restriction if the instance contains non-trivial intervals of measure zero for some players; otherwise the answer is unique. However, any such instance can be replaced by a sequence of instances with measures that are very close to the original ones and have non-zero density everywhere. If this is done carefully, all the α-points in the sequence of modified instances converge to the α-points in the original instance. Thus the restriction to a particularly chosen honest answer is not essential as well; on the other hand, it keeps the description of our lower bound much simpler.

3. The lower bound construction

In this section, we will prove the following theorem by means of an adversary argument in a decision tree. The adversary continuously observes the actions of the deterministic protocol, and he/she reacts by fixing the measures of the players appropriately.

**Theorem 3.1.** In the restricted cake cutting model of Section 2 (where each player is assigned a single interval), every deterministic fair cake division protocol for n players uses at least \( \Omega(n \log n) \) cuts and evaluation queries in the worst case.

Let us start by describing the specific cake measures \( \mu_p \) that we use in the input instances. Let \( \varepsilon < 1/n^4 \) be some small positive real number. For \( i = 1, \ldots, n \) we denote by \( X_i \subset [0, 1] \) the set consisting of the \( n \) points \( i/(n + 1) + k\varepsilon \) with \( 1 \leq k \leq n \). Moreover, we let \( X = \bigcup_{1 \leq i \leq n} X_i \). For \( p = 1, \ldots, n \), by definition the player \( p \) has his/her 0-point at position 0. The positions of the \( i/n \)-points with \( 1 \leq i \leq n \) are fixed by the adversary during the execution of the protocol: The \( i/n \)-points of all players are taken from \( X_i \), and distinct players receive distinct \( i/n \)-points. As one consequence, all the \( i/n \)-points of all players will lie strictly to the left of all the \( (i + 1)/n \)-points of all players.

All the cake value for player \( p \) is concentrated in tiny intervals \( I_{p,i} \) of length \( \varepsilon \) that are centered around his/her \( i/n \)-points: For \( i = 1, \ldots, n \), the measure of player \( p \) has a sharp peak with value \( i/(n^2 + n) \) immediately to the left of his/her \( i/n \)-point and a sharp peak with value \( (n - i)/(n^2 + n) \) immediately to the right of his/her \( i/n \)-point. Note that the measure between the \( i/n \)-point and the \( (i + 1)/n \)-point indeed adds up to \( 1/n \). Moreover, the measures of the two peaks around every \( i/n \)-point add up to \( 1/(n + 1) \), and the intervals that support these peaks for different players are always disjoint, with the exception of the intervals \( I_{p,0} \) that are the same for all the players. We do not explicitly describe the shape of the peaks; it can be arbitrary, but is determined in advance and the same for each player.

For every player \( p \), the portions of the cake between interval \( I_{p,i} \) and interval \( I_{p,i+1} \) have measure 0 and hence are worthless to \( p \). By our definition of \( \alpha \)-points, every \( \alpha \)-point of player \( p \) will fall into one of his/her intervals \( I_{p,i} \) with \( 0 \leq i \leq n \). If a player \( p \) cuts the cake at some point \( x \in I_{p,i} \), then we denote by \( c_p(x) \) the corresponding \( i/n \)-point of player \( p \).

**Lemma 3.2.** Let \( x \) be a cut that was done by player \( s \), and let \( y \geq x \) be another cut that was done by player \( t \). Let \( J = [x, y] \) and \( J' = [c_s(x), c_t(y)] \). If \( \mu_p(J) \geq 1/n \) for some player \( p \), then also \( \mu_p(J') \geq 1/n \).

**Proof.** Case 1. If \( s = p \) and \( t = p \), then let \( I_{p,j} \) and \( I_{p,k} \) be the intervals that contain the points \( c_p(x) \) and \( c_p(y) \), respectively. Then \( \mu_p(J) \geq 1/n \) implies \( k \geq j + 1 \). The measure \( \mu_p(J') \) is at least the measure \( (n - j)/(n^2 + n) \) of the peak immediately to the right of the \( j/n \)-point plus the measure \( k/(n^2 + n) \) immediately to the left of the \( k/n \)-point, and these two values add up to at least \( 1/n \).

Case 2. If \( s = p \) and \( t \neq p \), then let \( I_{p,j} \) be the interval that contains \( c_p(x) \). Then \( \mu_p(J) \geq 1/n \) implies that \( j < n \) and \( J \) and \( J' \) both contain \( I_{p,j+1} \), and again \( \mu_p(J') \) is at least \( 1/n \). Note that the argument works also if \( j = 0 \).
Case 3. The case \( s \neq p \) and \( t = p \) is similar to the second case above.

Case 4. If \( s \neq p \) and \( t \neq p \), then the interval between \( x \) and \( c_i(x) \) and the interval between \( y \) and \( c_i(y) \) both have measure 0 for player \( p \). By moving these two cuts, we do not change the value of \( \mathcal{J} \) for \( p \). ■

We call a protocol primitive if in all of its cut operations \( \text{Cut}(p; \alpha) \) the value \( \alpha \) is of the form \( i/n \) with \( 0 \leq i \leq n \).

**Lemma 3.3.** For every protocol \( \mathcal{P} \) in the restricted model, there exists a primitive protocol \( \mathcal{P}' \) in the restricted model, such that for every cake cutting instance \( I \) of the restricted form described above,

- \( \mathcal{P} \) and \( \mathcal{P}' \) make the same number of cuts on \( I \).
- if \( \mathcal{P} \) applied to instance \( I \) assigns to player \( p \) a piece \( \mathcal{J} \) of measure \( \mu_p(\mathcal{J}) \geq 1/n \), then also \( \mathcal{P}' \) applied to instance \( I \) assigns to player \( p \) a piece \( \mathcal{J}' \) of measure \( \mu_p(\mathcal{J}') \geq 1/n \).

**Proof.** Protocol \( \mathcal{P}' \) imitates protocol \( \mathcal{P} \). Whenever \( \mathcal{P} \) requests player \( p \) to cut at his/her \( \alpha \)-point \( x \) with \( 0 < \alpha < 1 \), then \( \mathcal{P}' \) computes the unique integer \( k \) with

\[
\frac{k}{n+1} < \alpha \leq \frac{k+1}{n+1}.
\]

Then \( \mathcal{P}' \) requests player \( p \) to cut the cake at his/her \( k/n \)-point. Note that by the choice of \( k \), this \( k/n \)-point equals \( c_p(x) \). The value of the cuts at \( x \) and \( c_p(x) \) is the same for all the players other than \( p \); thus any ensuing answer to an evaluation query is the same in \( \mathcal{P}' \) and \( \mathcal{P} \). Furthermore, since the shape of the peaks is predetermined and the same for all the players, from the cut of \( \mathcal{P}' \) at \( c_p(x) \) we can determine the original cut of \( \mathcal{P} \) at \( x \). Consequently \( \mathcal{P}' \) can simulate all the decisions of \( \mathcal{P} \). When assigning pieces, each original cut \( x \) of \( \mathcal{P} \) is replaced by the corresponding cut \( c_p(x) \) of \( \mathcal{P}' \). Clearly, both protocols make the same number of cuts, and Lemma 3.2 yields that if \( \mathcal{P} \) is fair, then also \( \mathcal{P}' \) is fair. ■

Hence, from now on we may concentrate on some fixed primitive protocol \( \mathcal{P}^* \), and on the situation where all cuts are from the set \( X \). The strategy of the adversary is based on a permutation \( \pi \) of the integers \( 1, \ldots, n \); this permutation \( \pi \) is kept secret and not known to the protocol \( \mathcal{P}^* \).

Now assume that at some point in time protocol \( \mathcal{P}^* \) asks player \( p \) to perform a cut at his/her \( i/n \)-point. Then the adversary fixes the measures as follows:

- If \( \pi(p) < i \), then the adversary assigns the \( i/n \)-point of player \( p \) to the smallest point in the set \( X_i \) that has not been used before.
- If \( \pi(p) > i \), then the adversary assigns the \( i/n \)-point of player \( p \) to the largest point in the set \( X_i \) that has not been used before.
- If \( \pi(p) = i \), then the adversary assigns the \( i/n \)-point of player \( p \) to the \( i \)th smallest point in the set \( X_i \).

Consequently, any possible assignment of \( i/n \)-points to points in \( X_i \) has the following form: The player \( q \) with \( \pi(q) = i \) sits at the \( i \)th smallest point. The \( i - 1 \) players with \( \pi(p) \leq i - 1 \) are at the first (smallest) \( i - 1 \) points, and the \( n - i \) players with \( \pi(p) \geq i + 1 \) are at the last (largest) \( n - i \) points. The precise ordering within the first \( i - 1 \) and within the last \( n - i \) players depends on the behavior of the protocol \( \mathcal{P}^* \). When protocol \( \mathcal{P}^* \) terminates, then the adversary fixes the ordering of the remaining \( i/n \)-points arbitrarily (but in agreement with the above rules).

**Lemma 3.4.** If \( \pi(p) \leq i \leq \pi(q) \) and \( p \neq q \), then in the ordering fixed by the adversary the \( i/n \)-point of player \( p \) strictly precedes the \( i/n \)-point of player \( q \).

**Proof.** This immediately follows from the adversary strategy above. ■

If the protocol \( \mathcal{P}^* \) asks a player \( p \) an evaluation query on an existing cut at the \( i/n \)-point of player \( p' \), then the current assignment of \( i/n \)-points to points in \( X_i \) and the permutation \( \pi \) determine whether the \( i/n \)-point of player \( p \) is smaller or larger than that of \( p' \) (for all the possible resulting assignments obeying the rules above). This is all that is necessary to determine the value of the cut, and thus the adversary can generate an honest answer to the query.

At the end, the primitive protocol \( \mathcal{P}^* \) must assign intervals to players: \( \mathcal{P}^* \) selects \( n - 1 \) of the cuts performed, say the cuts at positions \( 0 \leq y_1 \leq y_2 \leq \cdots \leq y_{n-1} \leq 1 \); moreover, we define \( y_0 = 0 \) and \( y_n = 1 \). Then for \( i = 1, \ldots, n \), the interval \([y_{i-1}, y_i]\) goes to player \( \phi(i) \), where \( \phi \) is a permutation of \( 1, \ldots, n \).
Lemma 3.5. If the primitive protocol $P^*$ is fair, then

(a) $y_i \in X_i$ for $1 \leq i \leq n - 1$.
(b) The interval $[y_{i-1}, y_i]$ contains the $(i-1)/n$-point and the $i/n$-point of player $\phi(i)$, for $1 \leq i \leq n$.

Proof. (a) If $y_1$ is at an $0/n$-point of some player, then $y_1 = 0$ and piece $[y_0, y_1]$ has measure 0 for player $\phi(1)$. If $y_{n-1} \in X_n$, then piece $[y_{n-1}, y_n]$ has measure at most $1/(n+1)$ for player $\phi(n)$. If $y_{i-1} \in X_j$ and $y_i \in X_j$ for some $2 \leq i \leq n-1$ and $1 \leq j \leq n-1$, then player $\phi(i)$ receives the piece $[y_{i-1}, y_i]$ of measure at most $1/(n+1)$. This leaves the claimed situation as the only possibility.

(b) Player $\phi(i)$ receives the cake interval $[y_{i-1}, y_i]$. By the statement in (a), this interval cannot cover player $\phi(i)$’s measure-peaks around $j/n$-points with $j < i - 1$ or with $j > i$. The two peaks around the $(i-1)/n$-point of player $\phi(i)$ yield only a measure of $1/(n+1)$; thus the interval cannot avoid the $i/n$-point. A symmetric argument shows that the interval cannot avoid the $(i-1)/n$-point of player $\phi(i)$. ■

Lemma 3.6. For any permutation $\sigma \neq \text{id}$ of the numbers $1, \ldots, n$, there exists some $i$ with $\sigma(i+1) \leq i \leq \sigma(i)$.

Proof. Take the minimum $i$ with $\sigma(i+1) \leq i$. ■

Finally, we claim that $\phi = \pi^{-1}$. Suppose otherwise. Then $\pi \circ \phi \neq \text{id}$ and by Lemma 3.6 there exists an $i$ such that $\pi(\phi(i+1)) \leq i \leq \pi(\phi(i))$.

Let $p := \phi(i+1)$ and $q := \phi(i)$, let $z_p$ denote the $i/n$-point of player $p$, and let $z_q$ denote the $i/n$-point of player $q$. Lemma 3.4 yields $z_p < z_q$. According to Lemma 3.5(b), point $z_p$ must be contained in $[y_i, y_{i+1}]$ and point $z_q$ must be contained in $[y_{i-1}, y_i]$. But this implies $z_p \geq y_i \geq z_q$ and contradicts $z_p < z_q$.

This contradiction shows that the assignment permutation $\rho$ of protocol $P^*$ must be equal to the inverse permutation of $\pi$. Hence, for each permutation $\pi$ the primitive protocol must reach a different leaf in the underlying decision tree. After an evaluation query Eval($p, x$), where $x$ is a result of Cut($p', i/n$), for $p \neq p'$ and $1 \leq i < n$, the protocol is returned one of two possible answers, namely $i/(n+1)$ or $(i + 1)/(n + 1)$, indicating whether Cut($p, i/n$) is before or after $x$ in $X_i$ (if $p = p'$ or $i \in [0, n]$, the answer is unique and trivial). After every query Cut($p, i/n$), the primitive protocol is returned one point of $X_i$: namely the first unused point if $\pi(p) < i$, the last unused point if $\pi(p) > i$, or the $i$th point if $\pi(p) = i$. Since the values in $X_i$ are known in advance, the whole protocol can be represented by a tree with a binary node for each possible evaluation query and a ternary node for each possible cut. The depth of a leaf in the tree is the number of cuts and evaluation queries performed for an instance corresponding to a given permutation. Since there are $n!$ permutations, the maximal depth of a leaf corresponding to some permutation must be at least $\log_3(n!) = \Omega(n \log n)$. This completes the proof of Theorem 3.1.

4. Discussion

One contribution of this paper is a discussion of various models and assumptions for cake cutting (that have appeared in the literature in some concise and implicit form) and a definition of a restricted model that covers the best protocols known. This discussion as well as our result show that evaluation queries play an important role, perhaps underestimated in previous literature. The lower bound of $\Omega(n \log n)$ on the number of cuts and evaluation queries needed for simple fair division with $n$ players is proved in a restricted model. The model clearly has its weak points (see, again, the discussion in Section 2), and it would be interesting to provide similar bounds in less restricted models. In particular, we suggest two open problems, related to the two restrictions in our model.

Problem 4.1. How many cuts are needed if no evaluation queries are allowed (but any player can be assigned several intervals)?

Our lower bound argument seems to break down even for ‘slight’ relaxations of the assumption about a single interval: On the instances from our lower bound, one can easily in $O(n)$ cuts assign to each player two of the intervals of size $\varepsilon$ that support his/her measure and this is clearly sufficient. And we do not even know how to make the lower bound work for the case where the cake is a circle, that is, for the cake that results from identifying the points 0 and 1 in the unit interval or equivalently when a single player can receive a share of two intervals, one containing 0 and one
containing 1. (Anyway, the circle is considered a non-standard cake and is not treated anywhere in the classical cake cutting literature [1,7].) Thus the restriction to a single subinterval share for each player seems very significant in our lower bound technique. On the other hand, all the protocols known to us obey this restriction.

Problem 4.2. How many cuts are needed if any player is required to receive a single subinterval (but evaluation queries are allowed and free)?

With evaluation queries, our lower bound breaks, since the decision tree is no longer ternary. After performing a cut, we may learn that \( \pi(p) < i \) or \( \pi(p) > i \), in which case we gain no additional information. However, once we find \( i \) such that \( \pi(p) = i \), the protocol finds out all values of \( p' \) satisfying \( \pi(p') < i \) and we can use recursion on the two subinstances. We can use this to give a protocol that uses only \( O(n \log \log n) \) cuts (and free evaluation queries) and works on the instances from our lower bound.

It would be very desirable to prove a lower bound for a model including free evaluation queries, or perhaps find some trade-off between cuts and evaluation queries. The perfectly fair protocols typically use only limited evaluations like “Is your measure of cake piece \( Z \) less than, greater than, or equal to the threshold \( \tau \)?” or “Is your measure of cake piece \( Z_1 \) less than, greater than, or equal to your measure of cake piece \( Z_2 \)?”. Perhaps handling these at first would be more accessible. We hope that this problem could be attacked by a similar lower bound technique using the decision trees in connection with a combinatorially richer set of instances.

Another interesting question concerns the randomized protocols. The randomized protocol of Even and Paz [2] uses an expected number of \( O(n) \) cuts and \( \Theta(n \log n) \) evaluation queries. Can the number of evaluation queries be decreased? Or can our lower bound be extended to randomized protocols?

Finally, let us remark that our model seems to be incomparable with that of Magdon-Ismail, Busch and Krishnamoorthy [4]. The set of instances for which they prove a lower bound of \( \Omega(n \log n) \) on the number of comparisons can be easily solved with \( O(n) \) cuts with no evaluation queries even in our restricted model. On the other hand, they prove a lower bound for protocols that have no restriction similar to our requirement of assigning a single subinterval to each player. The common feature of the two models seems to be exactly the lack of ability to incorporate the free evaluation queries; note that using an evaluation query generates at least one comparison.

Acknowledgements

The second author was partially supported by the Institute for Theoretical Computer Science, Prague (project LN00A056 of MŠMT ČR), and grant IAA1019401 of GA AV ČR.

References