Sub Additive Cake Cutting

Gorle Kunal Kaushik , *2019CS10352* **and Nallani Sai Venkata Kartheek,** *2019CS10377*
(kkgorle@gmail.com) (nsvk02@gmail.com)
supervised by **Prof. Rohit Vaish**

Abstract

We look at the problem of cake cutting with the goal of fairly allocating a divisible resource to agents with sub additive and monotone preferences. The key requirement is that each agent should be envy-free. The previous studies are primarily focused on additive preferences of agents which is not always the case in real life. For example, one wouldn't like more chocolates when they are full whereas they would like the same chocolate more when hungry. We'll provide the algorithms that give envy-free allocation for agents with such sub additive and monotone preferences. We further devise algorithms for computing envy-free allocations for bad cake under sub additive monotone preferences.

1 Introduction

Anything which holds some value for an agent is called Resource. A resource can be divisible like piece of land, time etc, in which any fractional division is possible or it can be indivisible like movie tickets, cricket balls where a fractional division is not possible over some extent.

1.1 Model and preferences

A divisible resource is also called a cake and is represented with the interval $[0,1]$. Every agent $i \in \{1,2,..,n\}$ will be having a valuation function v_i which gives his value for a piece of cake $\chi \subseteq [0,1]$, $v_i : \chi \longrightarrow \mathbb{R}^+$. χ need not be continuous. We also have the normalization condition which ensures that total value of cake for every agent is 1 i.e., $v_i([0,1]) = 1$

Additive Valuation : An agent i's preference is said to be additive if for any two disjoint pieces X , Y, $v_i(X \cup Y) = v_i(X) + v_i(Y)$.

Monotone Valuation : An agents valuation is said to be monotone if for any two pieces X , Y such that $X \subseteq Y$, $v_i(X) \leq v_i(Y)$. Additive valuations are monotone.

1.2 Fairness Notions

A partition $(A_1, A_2, .., A_n)$ of the cake, where A_i is a piece of the cake, distributed among n agents is called an allocation. The main requirement in such problems is fairness of the allocation like envy freeness or approximate envy freeness.

Envy Freeness : An allocation is said to be envy free if every agent weakly prefers his piece over any other agent's piece, $v_i(A_i) \geq v_i(A_j)$, $\forall i, j$.

Approximate Envy Freeness : An allocation is said to be approximately envy free or ϵ-envy free if no agent envies any other agent's piece more than a value of ϵ i.e., $v_i(A_i) \geq v_i(A_j) - \epsilon, \ \ \forall i, j$

1.3 Robertson Webb queries

For computational purposes of an cake cutting protocol, we use Robertson Webb Queries, Eval and Cut.

Eval : Eval query for an agent i computes it's value for a piece starting from a to b, $eval_i(a,b) \longrightarrow v_i([a,b])$.

Cut : Cut query for agent i takes a starting point and value as an input and returns the final point of the piece for which the value of the piece is as specified, $cut_i(a,\alpha) \longrightarrow b$ such that $v_i([a,b]) = \alpha$.

These queries are used as unit operations and considered $O(1)$ time and are used to give an estimate of time complexity for cake cutting algorithms.

1.4 Existing Results for additive valuations

There are different protocols for different n's that give envy free allocations.

- For two agents ($n = 2$), we have Cut and Choose protocol. In this protocol, one agent cuts the cake into his two equal pieces using cut query and the other picks his best using eval. This protocol takes two queries.
- For three agents ($n = 3$), we have Selfridge Conway Protocol. This takes five cuts and nine eval queries
- For three agents ($n = 3$), we also have Stromquist protocol. This uses moving knife procedure and is unbounded by Robertson Webb Query model.
- For any number of agents, Simmons Su Protocol uses sperners lemma and gives an allocation in which every agent have single continuous piece and this is unbounded by Robertson Webb Query Model.

- For any number of agents, Aziz and Mackenzie designed a bounded protocol for any n agents that gives envy free allocation with query complexity $O(n^{n^{n^{n^n}}})$.
- For any number of agents, we also have an algorithm that gives ϵ-EF allocation and uses envy cycle elimination method.

  In this method every agent cuts the cake into ϵ valued pieces. Now every piece in the cake is atmost ϵ for every agent and are considered goods. We run envy cycle elimination on these goods which returns an EF1 allocation (Every agent weakly prefers his bundle of goods over other agents bundle when removed one good from later) which is ϵ-EF as no piece is valued over ϵ for any agent.

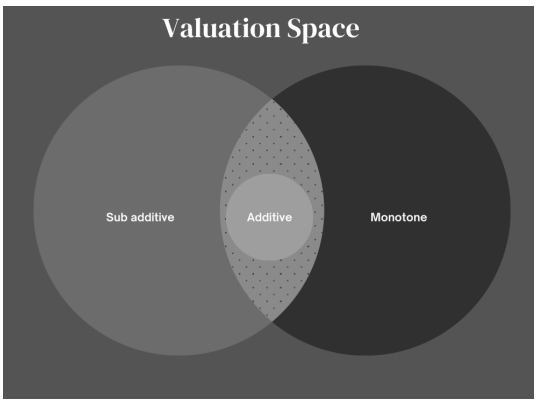
Table 1 shows all these results. (The connected pieces column in the table tells whether every agent's allocated piece is contiguous)

2 Our Problem

All the above and existing algorithms for envy freeness and ϵ-EF assume additive valuations. But this is not always true in real life cases like the pharmaceuticals or food division, etc., Many of these are sub additive in nature, like a person prefers same food, more when he is hungry, than when he is full.

Sub additive valuation : An agent i's preference is said to be sub additive, if for any two pieces X , Y, $v_i(X \cup Y) \leq v_i(X) + v_i(Y)$.

Our aim in this project is to design protocols that give fair allocations, for valuations that are sub additive and monotone, in finite time.



3 Challanges and New Queries

The traditional envy free cake cutting protocols do not easily extend to sub additive valuations. We need to define new queries and use new protocols for getting fair allocations in sub additive setting.

3.1 Equipartition Query:

Even for n=2 case, just with the existing eval and cut queries we will not be able to get an envy free allocation because cutting the cake into two equally valued pieces is not possible. (The first step of cut and choose protocol). If we use $cut_1(0, 1/2)$ then the right piece would have the value greater than 1/2 for agent 1 due to its subadditive nature. We wouldn't know at what boundary in cut query gives two equal halves of the cake. To solve this issue, we need a new query. We came up with *Equipartition Query* which divides a contiguous cake piece equally for a given agent into given number of pieces.

- *Equipartition Query* : *Divides a cake into given number of equal pieces for the given agent.*
- $Equipartition_i([a,b],k) \longrightarrow \{c_1, c_2, .., c_{k-1}\}$ where $v_i([a,c_1]) = v_i([c_1,c_2]) = ... = v_i([c_{k-2},c_{k-1}]) = v_i([c_{k-1},b])$.

  Effectively we are dividing the piece $[a,b]$ into k pieces which are of equal value for agent i. Equipartition into k pieces for given agent is always possible.

  Lemma 1: *The Equipartition into k pieces for given agent is always posssible.* (refer appendix for the proof).

3.2 Order Preserving condition on valuations

It has already been shown that for $n \geq 3$ no finite algorithm gives envy free allocation with connected pieces. Here arises the need to compare between values of unions of pieces in algorithms like Selfridge Conway. Selfridge Conway protocol involves dividing the cake into main and side cake and allocating a single piece from each of these cakes to each agent. In the Selfridge Conway Procedure every agent would be getting two contiguous pieces (say agent 2 gets pieces A & B , agent 3 gets pieces C & D in main and side cakes respectively where agent 1 is the primary cutter). In this case we have $v_2(A) \geq v_2(C)$ & $v_2(B) \geq v_2(D)$, which is guaranteed by selfridge conway procedure. In spite of having the above two guarantees, we still could not show $v_2(A \cup B) \geq v_2(C \cup D)$. Similar is the issue for agent 3. This is due to their sub additive nature in valuations that we are unable to compare unions. To solve this problem we enforce a new condition upon valuations called *Order Preserving.*

- *Order Preserving valuation :* A valuation is said to be Order Preserving if for any pieces W, X, Y, Z such that $W \cap X = \phi$ and $Y \cap Z = \phi$, if $v_i(W) \geq v_i(Y)$ & $v_i(X) \geq v_i(Z)$ then $v_i(W \cup X) \geq v_i(Y \cup Z)$. If both inequalities of the condition are weak, then final inequality is weak but if either one is strong then the final inequality is stronger i.e., if $v_i(W) > v_i(Y)$ & $v_i(X) \geq v_i(Z)$ then $v_i(W \cup X) > v_i(Y \cup Z)$.

Table 1. Table of Existing Results for additive valuations

Number of Agents	Protocol	Query Complexity	Connected Pieces
2	Cut and Choose	1 cut and 1 eval	Yes
3	Stromquist	∞	Yes
3	Selfridge Conway	5 cuts and 9 evals	No
n	Simmons-Su	∞	Yes
n	Aziz Mackenzie	$O(n^{n^{n^{n^n}}})$ queries	No
n	Envy cycle elimination method (ϵ-EF)	O(n/ϵ) cuts and O(n^3/ϵ) evals	No

This is a reasonable assumption to carry on with the inequality, that apply to individual pieces. One example of a sub additive monotone valuation that is also order-preserving is $v_i(W) = \sqrt{size(W)}$.

3.3 New Oracle (ϵ – oracle)

The time complexity of Aziz Mackenzie protocol that gives exact EF allocation for general n in additive valuations is super exponential. Many may prefer slightly relaxing fairness, to be satisfied with approximate EF or ϵ-EF allocation, to better the time complexity. This is where the envy cycle elimination method kicks in. In additive valuations, envy cycle elimination method works straight forward. Every agent cuts the cake into ϵ-valued pieces. Then we consider each piece as good and run envy cycle elimination to get EF1. This do not directly extend to the sub additive monotone valuations. The reason is that the first step of the algorithm, agents cutting cake into ϵ-valued pieces may take infinite number of cuts due to the sub additive nature of the valuation. (Let us say there are k cuts made by an agent, then sub additivity guarantess that $\epsilon \times k \geq 1$, which only implies $k \geq 1/\epsilon$). This makes the time complexity of the envy cycle elimination method for sub additive valuation to be infinite, which is not we want. To tackle this problem, we came up with a new oracle called **ϵ-oracle**, which ensures that time complexity is not infinite. This will be discussed in the theorem 3.

4 CONTRIBUTIONS

We have devised different protocols for getting fair allocations for sub additive monotone valuations. *(Table 2 shows these results).*

- For two agents (n=2), we used equipartition query to get EF allocation.
- For three agents (n=3), with condition of Order Preserving Valuations, we modified Selfridge Conway procedure to get EF allocation.
- For any number of agents, we used the new oracle (ϵ-oracle) to get ϵ-EF allocation.

Theorem 1: *There exist a envy-free allocation for two agents with sub additive monotone preferences, and it is given by Equipartition and choose protocol. This protocol takes 1 equipartition and 2 eval queries.*

Proof: Agent 1 partitions the cake into his two equal pieces using the query $\{x\} = Equipartition_1([0,1],2)$. Then agent 2 picks his best among two pieces checking $eval_2[0,x]$ and $eval_2[x,1]$ and agent 1 picks the remaining. This is clearly envy free as agent 2 is picking his best piece and agent 1 values both pieces equally. This protocol takes 1 Equipartition query and 2 eval queries.

Theorem 2: *There exist an envy-free allocation for three agents with sub additive, monotone and order preserving valuations. It is given by the Modified Selfridge Conway protocol. This protocol uses two equipartition queries, one cut and eleven evals.* (refer appendix for the proof)

Theorem 3: *There exist an ϵ-EF allocation for any number of agents with sub additive monotone preferences, and is given by the ϵ-oracle. This protocol takes $O(n^4/\epsilon)$ evals and $O(n^3/\epsilon)$ goodcuts.* (refer appendix for proof)

5 Moving from Good to Bad Cake

A resource need not always bring more value, it can reduce the value. If there exist a continuous resource which is hated by every agent, such resource is called bad cake. Cleaning a room, doing an assignment can be considered bad cakes.

5.1 Model and Preferences

Cake is represented with interval [0,1]. For any piece of cake, $\chi \subseteq [0,1]$, valuation for agent i is $v_i : \chi \longrightarrow \mathbb{R}^-$. Here also, χ need not be continuous. The normalisation condition here is $v_i([0,1]) = -1$

In this new setting of bad cake, the definitions of additive valuation and sub additive valuation remain same as that in good cake setting. The definition changes for monotone valuation.

Monotone Valuation : A valuation is said to be monotone if for any two pieces of cake X and Y such that $X \subseteq Y$, $v_i(X) \geq v_i(Y)$.

5.2 Existing Results

The fairness notions of EF and ϵ-EF also remain same as in good cake setting. There are traditional protocols for bad cakes that give fair allocations for additive valua-

Table 2. Results for sub additive monotone valuations

Number of Agents	Protocol	Query Complexity	Fairness
2	Equipartition and Choose	1 equipartition and 2 evals	EF
3	Modified Selfridge Conway (Works only on order preserving valuations)	2 equipartitions and 1 cut and 11 evals	EF
n	ϵ-oracle	$O(n^4/\epsilon)$ evals and $O(n^3/\epsilon)$ goodcuts	ϵ-EF

tions using the existing Robertson-Webb queries. (*Table 3 shows all these results*)

- For two agents (n=2), the same cut and choose protocol of the good cake, gives EF allocation.
- For three agents (n=3), a protocol called Reza-Oskui's protocol, which is similar but more complex than Selfridge Conway protocol, gives EF allocation.
- For any number of agents, Top Trading Envy Cycle elimination method, give ϵ-EF allocation.

  This method goes very similar to Envy Cycle elimination method of good cakes. Every agent cuts the cake into negative ϵ pieces, and each piece is considered chore, and top trading envy cycle elemination is done. This gives EF1 allocation which becomes ϵ-EF allocation of the cake.

5.3 Our Contributions

We looked at the problem of achieving fair allocations for bad cake when valuations are sub additive monotone.

5.3.1 Challenges

Again the protocols existing for additive valuations in bad cake do not directly extend. We have some challenges here as well. The challenges are very similar here with the challenges that we faced extending our results from additive valuations to sub additive valuations in good cake.

- The first problem of cutting a piece into equal valued halves is tackled by the equipartition query here also. The structure of the query is the same as in good cake.
- The second problem of inequalities not carrying to the unions, is tackled by the same order-preserving condition.
- The third problem is in Top Trading Envy Cycle elimination method. Here after the cuts happen, and top trading envy cycle elemination algorithm is run, we get EF1 allocation. This EF1 allocation will not be the same as the $epsilon$-EF in cake cutting due to the sub additive nature of valuation. So this algorithm does not work. We need a new approach. So we devised a new oracle called **negative-ϵ-oracle** for finding ϵ-EF allocation for any general n, for the bad cake. This will be discussed in theorem 6.

5.3.2 Results

We have devised different protocols for getting fair allocations for sub additive monotone valuations for the bad cake. *(Table 4 shows these results).*

Theorem 4: *There exist an envy-free allocation for two agents with sub additive monotone preferences in bad cake setting. It is given by equipartition and choose protocol. This protocol uses 1 equipartition and 2 eval queries.*

Proof: The algorithm is same as in good cake. Agent 1 partitions the cake into his two equal pieces using the query $\{x\} = Equipartition_1([0,1],2)$. Then agent 2 picks his best among two pieces checking $eval_2[0,x]$ and $eval_2[x,1]$ and agent 1 picks the remaining. This is clearly envy free as agent 2 is picking his best piece and agent 1 values both pieces equally. This protocol takes 1 Equipartition query and 2 eval queries.

Theorem 5: *There exist an envy-free allocation for three agents with sub additive, monotone and order preserving valuations in bad cake setting. It is given by the Modified Reza Oskui's protocol. This protocol uses three equipartition queries, four cut and eighteen evals.* (refer appendix for the proof)

Theorem 6: *There exist an ϵ-EF allocation for any number of agents with sub additive monotone preferences in the bad cake setting. It is given by the negative-ϵ-oracle. This protocol takes $O(n^3/\epsilon)$ evals and $O(n^2/\epsilon)$ badcuts query complexity.* (refer appendix for proof)

6 Conclusion:

There were protocols to get fair allocations, envy free or ϵ-envy free, both for good and bad cake, for different number of agents in finite time in additive valuation setting. But when we move to sub additive and monotone valuation setting, these results do not easily extend. There is need of defining new queries and having some constraints on valuations. We defined 'equipartition' query, 'goodcut' and 'badcut' queries. We also constrained the valuations to be order-preserving, which was reasonable.

Using these new queries and valuation constraints, we showed the existence of protocols that gives exact envy free allocations for two agents and three agents, both for good and bad cakes. We also showed protocols that give approximate envy free (or ϵ-EF) allocation for any number of agents, both for good and bad cakes.

Table 3. Existing Results for bad cake for additive valuations

Number of Agents	Protocol	Query Complexity	Fairness
2	Cut and Choose	1 cut and 1 eval	EF
3	Reza Oskui	10 cuts and 14 evals	EF
n	Top Trading Envy Cycle elimination method	O(n/ϵ) cuts and O(n^3/ϵ) evals	ϵ-EF

7 Future Research:

One possible future research is to get exact envy free allocation for any number of agents i.e, extending Aziz-Mackenzie protocol to sub additive monotone valuations. Other direction is to improve the results in this paper using lesser number of queries or using more simpler queries. Working with mixed cake is also a research possibility. One can also work with other fairness notions. The research done on this setting of cake cutting is still very little, and has a big scope of research possible.

8 References:

- The 'Cake Cutting Algorithms' chapter from 'Handbook of Computational Social Science' by Felix Brandt
- Lecture on 'Cake Cutting' by Prof. Rohit Vaish in 'Algorithms for Collective Decision-Making' course
- The 'Cake Cutting Algorithms - Be fair if you can' book by Jack Robertson and Willian Webb
- Walter Stromquist's 2008 paper on 'Envy-free cake divisions cannot be found by finite protocols'

APPENDIX

Lemma 1: *The Equipartition into k pieces for given agent is always posssible.* (refer appendix for the proof).

Proof: We could prove this using mathematical induction. We can consider base case to be k=2. We can use a moving knife to prove this. We would consider the values of pieces left of knife, and right of knife. When, the knife is at left most, left piece have value 0 and right piece have value 1. When knife is at its right most, the vice versa happens. So, by intermediate value Theorem, there is a position of knife, that left and right pieces have equal value and the query returns this value. So, the base case proved.

Let us assume the inductive hypothesis, equipartition of a cake into k pieces exist. It is enough for us to prove there exist a equipartition of cake into k+1 pieces. Again we use the moving knife procedure. But at every position of cake, we equipartitions the right piece into k pieces (using hypothesis). We consider the left piece value and one among the partitioned values in the right piece. Initilly when cake is at left most, left piece have value 0 and partitioned right piece have some value atleast 1/k (guaranteed by sub additive property). Then when the knife is at right most end, left piece have value 1 and right side partioned piece have value 0. Again by intermediate value Theorem, we can say that there is a position of knife, where the left piece values equal to any partitioned right side piece. Here effectively, the cake is partitioned into k+1 equal pieces, which proves induction step. Hence, by mathematical induction, one can say that equipartition into any k pieces for a given agent always exist.

Theorem 2: *There exist an envy-free allocation for three agents with sub additive, monotone and order preserving valuations. It is given by the Modified Selfridge Conway protocol. This protocol uses two equipartition queries, one cut and eleven evals.* (refer appendix for the proof)

Proof: The algorithm goes very similar to the traditional selfridge conway protocol. The only steps that are different here are we use Equipartition queries for cutting a piece into three equal pieces rather than using the usual cut queries. In the algorithm, first agent 1 partitions the whole cake into three equal pieces using the query $\{x, y\} = Equipartition_1([0,1],3)$ [1 equipartition query]. Then agent 2 points to his first and second favourite pieces among $A = [0, x], B = [x, y], C = [y, 1]$ [using three eval queries]. Let them be A, C respectively. Agent 2 cuts out piece from A [using 1 cut query] so that remaining piece value equals to that of piece C (makes a two-way tie for the favourite). Let the cut out piece be called side piece, S. The remaining piece is called the main cake, $M = A' \cup B \cup C \ where, A' = A - \{S\}$).

Now we allocate pieces M and S to all three agents. In main cake, we ask agents to pick their best piece among A', B, C in the order of agents 3 [take 3 evals], 2, 1. We also force agent 2 to pick A', if agent 3 does not pick it. Let the agent that picked A' be T. T could either be agent 3 or it could be agent 2. Let T' be the agent, that is not T, among $\{agent\,2, agent\,3\}$.

For allocating side cake, T' first partitions S into three equal pieces [1 equipartition query]. Then agents pick the three pieces of S in the order T [3 evals], agent 1 [2 evals], T'.

In this allocation, the partial allocaion of M is envy free because agent 3 is picking his best, agent 2 is getting one of his two tied favourites, and agent 1 is surely getting a non trimmed piece (B or C), which is atleast as big as other two. In side piece T' and T are envy free because T' is cutting the cake into his three equal pieces and T is picking his best. Agent 1 also does not have envy towards T as he picks earlier. One could see that in overall, T and T' are envy free. This is guaranteed by Order preserving

Table 4. Table of Results for bad cake for sub additive monotone valuations

Number of Agents	Protocol	Query Complexity	Fairness
2	Equipartition and Choose	1 equipartition and 2 evals	EF
3	Modified Reza Oskui (Works only on order preserving valuations)	3 equipartitions and 4 cuts and 18 evals	EF
n	negative-ϵ-oracle	$O(n^3/\epsilon)$ evals and $O(n^2/\epsilon)$ badcuts	ϵ-EF

condition as they both individually are envy free in both main and side cakes. Similar argument follows for agent 1 being non-envious of T. The only final envy that could be possible is for agent 1 towards T'. Even this envy does not exist because the overall piece of T' will be a sub-piece of A, which has no more value for agent 1 than the piece he gets in main cake (as he had cut the cake into his three equal pieces and monotoneity ensures the claim). So no agent envies any other which makes the allocation envyfree.

Theorem 3: *There exist an ϵ-EF allocation for any number of agents with sub additive monotone preferences, and is given by the ϵ-oracle. This protocol takes $O(n^4/\epsilon)$ evals and $O(n^3/\epsilon)$ goodcuts.* (refer appendix for proof)

Proof: This oracle involves a new query. The goodcut query is defined like this, $goodcut_i(a, X_i, X_s, \epsilon)$ returns b, where $v_i(X_s \cup [a,b]) = v_i(X_i) + \epsilon$. It returns null, if such b doesn't exist. The oracle is as follows. At every step, we locate a source s, and run goodcut queries with all other agents $i \neq s$, with X_s being current bundle of source s and X_i being current bundle of agent i and 'a' being the starting point of un-allocated cake. Then we take the smallest among all b's that goodcut queries return and allocate [a,b] to the source. We then update the envy graph and resolve if there is a cycle to get a source, then update 'a' value to 'b' and continue with the same step. Firstly, this algorithm gives ϵ-EF allocation. This is because in every step, we allocate cake to source agent s, until every other agent can envy the former with value not more than ϵ. This ensures ϵ-EF allocation at every step, which means final allocation is ϵ-EF.

We should also ensure the finiteness of the protocol. For that we define $\phi_X = \Sigma_i \Sigma_j v_i(X_j)$, where X is an allcotaion and X_j is piece of agent j in X. For an initial empty allocation X, $\phi_X = 0$ and for comple allocation, $\phi_X \leq n^2$ due to sub additivity (every $v_i(j) \leq 1$). Now at every step let us look how much does the ϕ increase. Let allocation of the cake before a step is X afterwards is X'. Let source s got a piece [a,b] in that step and agent i's goodcut is the smallest. We calculate $\phi_{X'} - \phi_X$. This would be $\Sigma_j(v_j(X'_s) - v_j(X_s))$, as only allocation of agent s is changing in the step. This term would be $\Sigma_{j \neq i}(v_j(X'_s) - v_j(X_s)) + (v_i(X'_s) - v_i(X_s))$. The first term is always non-negative due to monotone property (as $X'_s = X_s \cup [a,b]$) and second term values exactly ϵ according to goodcut query. So, we get $\phi_{X'} - \phi_X \geq \epsilon$. That is ϕ increases atleast by ϵ in every step, which makes maximum number of steps to be n^2/ϵ. So algorithm terminates and finiteness was guaranteed. As we are having, $O(n^2/\epsilon)$ steps, alorithm takes $O(n^3/\epsilon)$ goodcut queries (n goodcuts in each step) and $O(n^4/\epsilon)$ eval queries (envy graph updating of $O(n^2)$ evals in every step). So, total query complexity is $O(n^3/\epsilon)$ goodcut queries and $O(n^4/\epsilon)$ eval queries.

Theorem 5: *There exist an envy-free allocation for three agents with sub additive, monotone and order preserving valuations in bad cake setting. It is given by the Modified Reza Oskui's protocol. This protocol uses three equipartition queries, four cut and eighteen evals.* (refer appendix for the proof)

Proof: This protocol has some similarity with the modified Selfridge Conway protocol. Agent A (let the agents be A, B, C) first equi-partitions [1 equipartition] the cake into three pieces, call them X_1, X_2, X_3. If agents B and C has different favourite piece, then we just can allocate their favourite piece and allocation would be envy-free. But if they have same favourite piece [3 evals for B and 3 for C] (say X_1), then we go with the following procedure. Agents B and C marks their cuts on X_2 and X_3 [2 cuts for B and 2 for C] so that the remaining pieces would have value for them equal to that of X_1. The agents who's trim is weaker (agent who has to remove smaller piece) will cut the respective piece. Here three cases arise.

Case 1: Agent B's trims are weaker in both X_2 and X_3. In this case, B trims X_2 and X_3 into X'_2 and X'_3. Let the cut out pieces be E_2 and E_3. So, main cake M would be $X'_2 \cup X_1 \cup X'_3$. The side cakes would be E_2 and E_3. Now in the main cake, we can assign X_1 to agent C, A takes his best among X'_2 and X'_3 [2 eval queries]. B takes the remaining. (say A gets X'_2 and B gets X'_3). This partial allocation is envy free as A is getting better than X_1 due to the monotoneity; for B, all the pieces in the main cake are of equal value; and for C, X_1 is the best as the trims were longer for C in X_2 and X_3. So, the partial allocation in main cake is envy free. Coming to E_2 and E_3, agent B equipartitions both into three pieces each [2 equipartitions]. The picking order would be C, A, B in both E_2 and E_3 [3 evals for C and 2 for B in each piece]. In both the side cakes E_2 and E_3, agent C won't envy anyone as he picks first. Agent B won't envy any other because he is the divider of both pieces. Agent A won't envy B as he is picking ahead. Now the only possible envy possible is A toward C. We will prove that is not possible. Let the pieces A, B, and C get in E_2 and E_3 are $\{a_2, b_2, c_2\} and \{a_3, b_3, c_3\}$ respectively. The final allocation of C is $X_1 \cup c_2 \cup c_3$ and A is $X'_2 \cup a_2 \cup a_3$. We have to prove $v_A(X'_2 \cup a_2 \cup a_3) \geq v_A(X_1 \cup c_2 \cup c_3)$. For this proof, I first claim that $v_A(a_3) \geq v_A(b_2 \cup c_2)$. We can prove this by method of contradiction. We first assume $v_A(a_3) < v_A(b_2 \cup c_2)$. This implies that $v_A(a_2 \cup a_3) < v_A(a_2 \cup b_2 \cup c_2) = v_A(E_2)$ (as valuations are order preserv-

ing). We also have $v_A(b_2 \cup b_3) \leq v_A(a_2 \cup a_3)$ (as agent A picks before B in both E_2 and E_3) and $v_A(E_2) \leq v_A(E_3)$ (as A prefers X_2' over X_3', order preserving guarantees he prefer E_3 over E_2 because A values $E_2 \cup X_2', E_3 \cup X_3' equally$). Substituting last two equaitions in the equation before gives $v_A(b_2 \cup b_3) < v_A(E_3) = v_A(a_3 \cup b_3 \cup c_3)$. This implies that $v_A(b_2) < v_A(a_3 \cup c_3)$ (according to order preserving condition). Now this means $v_A(b_2 \cup c_2) < v_A(a_3 \cup c_3 \cup c_2)$ (again according to order preserving condition). But we already have $v_A(a_3) < v_A(b_2 \cup c_2)$, so this means $v_A(a_3) < v_A(a_3 \cup c_3 \cup c_2)$. This is a contradiction as valuations are monotone. So our assumption of $v_A(a_3) < v_A(b_2 \cup c_2)$ is wrong. So therefore $v_A(a_3) \geq v_A(b_2 \cup c_2)$. This means $v_A(a_2 \cup a_3) \geq v_A(a_2 \cup b_2 \cup c_2) = v_A(E_2)$. This means $v_A(X_2' \cup a_2 \cup a_3) \geq v_A(X_2' \cup E_2) = v_A(X_2) = v_A(X_1)$. This means $v_A(X_2' \cup a_2 \cup a_3) \geq v_A(X_1 \cup c_2 \cup c_3)$ as valuation is monotone. So, A is envy free of C. So here any agent does not envy other agents. So, allocation is envy free in this case.

Case 2: Agent C's trims are weaker in both X_2 and X_3. Then the proof goes in very similar way to that of case 1. Only agents B and C will be exchanged. So, in this case also, we get the final allocation envy free.

Case 3: Agent B's trim is weaker in one of X_2 and X_3 and C's trim is weaker in the other (say B trims X_2 and C trims X_3). In this cake, in main cake, agent A gets his best piece among X_2' and X_3'. Agent B picks X_2' if A did not pick it, otherwise he picks X_1. Agent C picks X_3' if A did not pick it, otherwise he picks X_1. This partial allocation of main cake is envy free as A is getting his best, whereas B and C are getting one of their two-way tied favourites. Both the side cakes gets divided by one of Agent B and Ageent C, who did not get piece X_1. The rest of the proof goes in same way as in case 1. So, here also the final allocation is envy free.

Therefore, the allocation that the modifed reza-oskui's protocol gives is envy free and it takes 3 equipartition, 4 cut and 18 eval queries.

Theorem 6: *There exist an ϵ-EF allocation for any number of agents with sub additive monotone preferences in the bad cake setting. It is given by the negative-ϵ-oracle. This protocol takes $O(n^3/\epsilon)$ evals and $O(n^2/\epsilon)$ badcuts query complexity.* (refer appendix for proof)

Proof: This oracle involves a new query. The badcut query is defined like this, $badcut_s(a, X_i, X_s, \epsilon)$ returns b, where $v_s(X_s \cup [a,b]) = v_s(X_i) - \epsilon$. It returns null, if such b doesn't exist. The oracle is as follows. At every step, we locate a sink s, and run badcut queries for sink s with all other agents $i \neq s$, with X_s being current bundle of sink s and X_i being current bundle of agent i and 'a' being the starting point of un-allocated cake. Then we take the smallest among all b's that badcut queries return and allocate [a,b] to the sink. We then update the envy graph and resolve if there is a cycle to get a sink, then update 'a' value to 'b' and continue with the same step. Firstly, this algorithm gives ϵ-EF allocation. This is because in every step, we allocate cake to sink agent s, until agent s does not envy any other agent with a value more than ϵ. This ensures ϵ-EF allocation at every step, which means final allocation is ϵ-EF.

We should also ensure the finiteness of the protocol. For that we define $\phi_X = \Sigma_i \Sigma_j v_i(X_j)$, where X is an allcotaion and X_j is piece of agent j in X. For an initial empty allocation X, $\phi_X = 0$ and for comple allocation, $\phi_X \geq -n$ due to sub additivity (for every i, $\Sigma_j v_i(X_j) \geq v_i([0,1]) = -1$). Now at every step let us look how much does the ϕ decrease. Let allocation of the cake before a step is X afterwards is X'. Let sink s got a piece [a,b] in that step and badcut of s with agent i is the smallest. We calculate $\phi_{X'} - \phi_X$. This would be $\Sigma_j(v_j(X_s') - v_j(X_s))$, as only allocation of agent s is changing in the step. This term would be $\Sigma_{j \neq s}(v_j(X_s') - v_j(X_s)) + (v_s(X_s') - v_s(X_s))$. The first term is always non-positive due to monotone property (as $X_s' = X_s \cup [a,b]$) and second term is at most $v_s(X_s') - v_s(X_i)$ (as s is sink in allcoation X), and this term is exactly negative-ϵ according to badcut query. So, we get $\phi_{X'} - \phi_X \leq -\epsilon$. That is ϕ decreases atleast by ϵ in every step, which makes maximum number of steps to be n/ϵ. So algorithm terminates and finiteness was guaranteed. As we are having, $O(n/\epsilon)$ steps, alorithm takes $O(n^2/\epsilon)$ badcut queries (n badcuts in each step) and $O(n^3/\epsilon)$ eval queries (envy graph updating of $O(n^2)$ evals in every step). So, total query complexity is $O(n^2/\epsilon)$ badcut queries and $O(n^3/\epsilon)$ eval queries.