Confidently Cutting a Cake into Approximately Fair Pieces

Jeff Edmonds[1], Kirk Pruhs[2*], and Jaisingh Solanki[1]

[1] Department of Computer Science and Engineering
York University
jeff@cse.yorku.ca

[2] Computer Science Department
University of Pittsburgh
kirk@cs.pitt.edu

Abstract. We give a randomized protocol for the classic cake cutting problem that guarantees approximate proportional fairness, and with high probability uses a linear number of cuts.

1 Introduction

The classic cake cutting problems originated in the 1940's in the Polish mathematics community and involves fairly apportioning valuable resources (the cake) when there is not an agreed upon value of the resources. The analogy of cake was used because of the well known phenomenon of people valuing frosting and cake differently. Cake cutting is widely studied within the social sciences because of the obvious importance of fairly dividing resources, and is widely studied in the mathematical sciences because of the elegance of the problems. There are several books written on cake cutting, and related fair allocation problems (See, for example, [3, 12]). Because of the inherent interest of cake cutting problems to a wide audience, cake cutting is often taught in discrete mathematics courses, and often appears in the media in shows that try to popularize mathematics. For example, in the "One Hour" episode of the TV show Numb3rs, the lead FBI agent uses his understanding of cake cutting algorithms to deduce the portion of the ransom received by the head of a kidnapping conspiracy. [3]

The setting for the cake cutting problem involves a continuous resource modeled by the unit interval, n players, a value function V_p for each player p, and a referee protocol. The value function for each player specifies how much that player values each subinterval of the cake. A piece is a union of disjoint subintervals, and the value function is additive, so that the value of a piece is the sum of the values of the underlying subintervals. The value functions are initially unknown to the referee. The standard operation is a cut query, in which the referee asks the player to identify the shortest subinterval with a fixed value and a fixed left endpoint. We assume here that the players answer queries honestly (for more discussion of this issue, see section 2). After the cut

[*] Supported in part by NSF grants CNS-0325353, CCF-0514058 and IIS-0534531.

[3] Although, as is often the case in this show, the application of mathematics is designed to aid the plot development, and is not necessarily designed to satisfy mathematician's sensibilities.

queries, the referee partitions the resulting subintervals among the players. The referee's goal is to fairly apportion the cake among the players. There are several notions of fairness in the literature, but the original, and most basic, notion is that of proportional fairness. An apportionment is *proportionally fair*, or simply *proportional* if each player believes that his piece is worth at least $1/n$ of the total value of the cake, according to that player's value function. In this paper, we consider the notion of approximate fairness. We will say that an apportionment is *c-fair* fair if each player believes that his piece is worth at least $1/(cn)$ of the total value of the cake, according to that player's value function. We will say that a referee protocol is *approximately fair* if there exists some constant $c \geq 1$ such that the protocol guarantees a c-fair apportionment. We are interested in the query complexity of a referee protocol, which is the worst-case number of queries required to achieve a fair allocation for each player.

A deterministic proportional protocol with query complexity $\Theta(n^2)$ was given in 1948 by Steinhaus in [14]. In 1984, Even and Paz [6] gave a deterministic divide and conquer proportional protocol that has query complexity $\Theta(n \log n)$. Recently, there has been several papers [7, 13, 4] that give lower bounds on the query complexity for proportional cake cutting. Sgall and Woeginger [13] showed that every proportional protocol (deterministic or randomized) has query complexity $\Omega(n \log n)$ if each player must receive a contiguous piece of the cake. Edmonds and Pruhs [4] show that the query complexity of every deterministic approximately-fair protocol is $\Omega(n \log n)$.

This left open the question of whether approximate-fairness was achievable by a randomized protocol with query complexity $O(n)$. In the subsequent paper [5], Edmonds and Pruhs was settled this question in the affirmative by giving a randomized approximately-fair protocol with expected query complexity $O(n)$. This protocol was based on the following theorem:

Lemma 1 (Balanced Allocation Lemma [5]). *Let $\alpha \geq 17$ be some sufficiently large constant. Assume each of the n players conceptually partitions the unit interval into αn disjoint candidate subintervals/pieces of equal value. Then assume that each player independently picks $d' = 2d = 4$ of her candidate pieces uniformly at random, with replacement. Then there is an efficient apportionment method that, with probability $\Omega(1)$, assigns each player one final piece of her d' candidate pieces, so that every point on the cake is covered by at most 2 players.*

Once overlap of $O(1)$ is achieved for every point of the cake, one can achieve approximate fairness with linearly more queries by using any proportional algorithm to apportion the portions of cake where there is contention among the final pieces. By applying the Edmonds and Pruhs Balanced Allocation Theorem until a successful apportionment is possible, one get expected query complexity of $O(n)$. This is because each application of the Balanced Allocation Theorem has complexity $O(n)$, and the number of applications until a success is a geometrically distributed random variable with probability of success $\Omega(1)$.

However, several referees of [5] complained that in order to have high confidence of success one would have to apply the Balanced Allocation Theorem $\Omega(\log n)$ times, and thus if one accepts the requirement that a randomized algorithm should succeed with high probability, then this randomized algorithm would have no better query complexity than the Evan and Paz deterministic algorithm. In the context of this paper, high probability of success means that the probability of failure, as a function of n, should approach zero as n increases.

In this paper, we answer the referees of [5] by showing the following high confidence version of the Balanced Allocation Lemma in [5].

Lemma 2 (High Confidence Balanced Allocation Lemma). *Let $\alpha \geq 17$ be some sufficiently large constant. Assume each of the n players conceptually partitions the unit interval into αn disjoint candidate subintervals of equal value. Each player independently picks $d' = k \times 2 \times 2d = 8k$ of her pieces uniformly at random, with replacement. Then there is an efficient method that, with probability $(1 - O(\frac{1}{n^k}))$, picks one of the d' subintervals for each player, so that every point on the cake is covered by at most 4 players.*

The rest of the paper is organized as follows. In section 2 we discuss some other related results, and explain how the standard balls and bins model is a special case of cake cutting. In section 3 we briefly explain how [5] obtained a (low confidence) proof of the Balanced Allocation Lemma for cake. In section 4 we give a protocol, which turns out to be a rather simple modification of the protocol in [5], that establishes the High Confidence Balanced Allocation Lemma. We discovered the simple protocol in section 4 after much effort on another approach. We briefly discuss this other approach in section 5 because we believe that it poses some interesting open questions.

2 Other Related Work

The lower bound proofs in [13, 4] also allow the referee to make evaluation queries, which ask a player to state their value for a particular piece. Edmonds and Pruhs [4] also showed that every randomized approximately-fair protocol has query complexity $\Omega(n \log n)$ if answers to the queries asked by protocol are approximations to actual answers. Approximately fair protocols were introduced by Robertson and Webb [11]. There is deterministic protocol [11, 10, 15] that achieves approximate-fairness with $\Theta(n)$ cuts and $\Theta(n^2)$ evaluations. There are several other notions of fairness studied in the cake cutting setting, most notably envy-free fairness. There are known finite complexity protocols for envy-free divisions, but no bounded complexity protocols are known (See, for example, [3] for details). The cake cutting problem is often defined so that the players do not need to answer the queries truthfully. For deterministic protocols, lying is generally a non-issue since it is easy to catch any form of lying that would mess up the standard protocols (See [3] for details). But for randomized protocols, it seems much more difficult to catch cheaters.

In the multiple-choice balls and bins model, d of αn discrete bins are selected for each ball uniformly at random. Then we select one bin out of d bins such that maximum number of balls in any bin is minimized. There is an efficient procedure, essentially a matching algorithm for a bipartite graph, that picks one of the d bins for each player so that maximum number of balls in any bin is $O(1)$, with probability $(1-O(\frac{1}{poly(n)}))$. [2]. The balls and bins model is equivalent to the special case of the cake model in which all the players value the cake uniformly. Analysis of the balls and bins model has found wide applications in areas such as load balancing [8]. In these situations, a ball represents a job that can be assigned to various bins/machines/servers. Roughly speaking, load balancing of identical machines is to balls and bins, as load balancing on unrelated machines is to cake cutting. In the unrelated machine model, the speed that a machine runs a job depends on the job. So the jobs may not agree on the values of the various machines. Unrelated machines is one of the standard models in the load balancing literature [1].

In the balls and bins model, the maximum number of balls in any bin is $\theta(\frac{\log n}{\log \log n})$ with probability $(1 - O(\frac{1}{poly(n)}))$ [9]. Assume n balls are thrown sequentially into n bins, each ball is placed in the least full bin at the time of the placement, among d bins, $d \geq 2$, chosen independently and uniformly at random. Then after all the balls are placed, the maximum number of balls in any bin is $\theta(\frac{\log \log n}{\log d})$ with probability $\Omega(1 - \frac{1}{poly(n)})$. [2].

3 The Original Balanced Allocation Lemma for Cake

We now outline the protocol and analysis that establishes the Balanced Allocation Lemma in [5]. Note this protocol uses two graphs, the implication graph, and the same-player-vee graph, and some graph theoretic definitions, that we will define after the protocol.

Edmonds-Pruhs Protocol:

– **Step 1:** Independently, for each player $p \in [1, n]$ and each $r \in [0, 1]$, randomly choose $d = 2$ of the candidate pieces $c_{\langle p,i \rangle}$ to be in the quarterfinal bracket $A_{\langle p,r \rangle}$. Thus each player has two quarterfinals brackets, each containing two intervals.
– **Step 2:** In each quarterfinal bracket $A_{\langle p,r \rangle}$, pick as the semifinal piece $a_{\langle p,r \rangle}$, the piece that intersects the fewest other candidate pieces $c_{\langle q,j \rangle}$. Thus each player is left with two semifinal intervals.
– **Step 3:** Form the implication graph and same-player-vee graph for the semifinal pieces
– **Step 4:** If implication graph contains a pair path of length greater than or equal to 3, then admit failure.
– **Step 5:** If same-player-vee graph is not $w = 2$ colorable, then admit failure.
– **Step 6:** Let S_h be the subgraph of the implication graph containing only those players colored h, in the same-player-vee graph. This ensures that implication graph restricted to S_h contains no pair paths of length 2.
– **Step 7:** For each S_h, pick the final piece for each player involved in S_h by applying the Final Piece Selection Algorithm to S_h.

We now turn to the graph theoretic notions used in this protocol. The vertices of the *implication graph* IG are the $2n$ pieces $a_{\langle p,r \rangle}$, $1 \leq p \leq n$ and $0 \leq r \leq 1$, and if piece $a_{\langle p,r \rangle}$ intersects piece $a_{\langle q,s \rangle}$, then there is a directed edge from piece $a_{\langle p,r \rangle}$ to piece $a_{\langle q,1-s \rangle}$, and similarly from $a_{\langle q,s \rangle}$ to $a_{\langle p,1-r \rangle}$. The intuition behind the this definition is that if a player p gets $a_{\langle p,r \rangle}$ as her final piece, then player q must get piece $a_{\langle q,1-s \rangle}$ if p's and q's pieces are not to overlap. Similarly if q gets $a_{\langle q,s \rangle}$, then p must get $a_{\langle p,1-r \rangle}$. As an example, Figure 1 gives a subset of the semifinal pieces selected from the candidate pieces. The corresponding implication graph is also given in Figure 1. A *pair path* in an implication graph is a directed path between two pieces for one player.

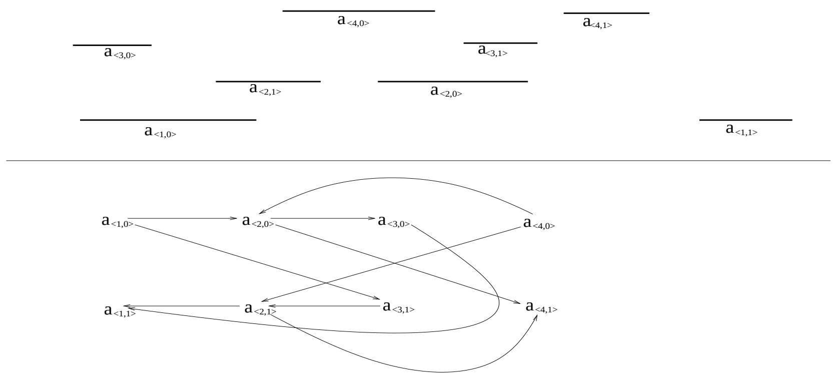

Fig. 1. Players' two selected pieces and corresponding implication graph.

In Figure 1, there are two pair paths of length three from the first player's left semifinal piece to her right and two pair paths of length two from the fourth player's left semifinal piece to her right. Pair paths are problematic because they effectively imply that if the first player gets her left semifinal piece as his final piece then she must get her right piece too. Edmonds and Pruhs [5] prove that if the implication graph IG does not contain pair paths then the following algorithm selects a final piece for each player in such a way that these final pieces are disjoint.

Final Piece Selection Algorithm: We repeatedly pick an arbitrary player p that has not selected a final piece. We pick the piece $a_{\langle p,0\rangle}$ as the final piece for p. Further, we pick as final pieces all those pieces in IG that are reachable from $a_{\langle p,0\rangle}$ in IG.

If edges in the implication graph were independent, then we could bound the probability of a pair path as in the balls and bins case:

$$\text{Prob}[IG\ contain\ pair\ paths] \approx \sum_{z=2}^{n} \binom{n}{z} \left(\frac{1}{\alpha n}\right)^{z} \approx \frac{1}{\alpha(\alpha-1)}.$$

Unfortunately, edges in the implication graph are not statistically independent. For example, if all of player B and player C's pieces are contained in one candidate piece P for player A, then the existence of an edge involving A and B would mean that player A picked candidate piece P, and thus thus there must be an edge involving player A and player C. Nevertheless, [5] show that, in spite of the statistical dependencies of edges, the above calculation of the probability of a pair path does give approximately the right answer.

First, [5] observed the vital difference between pair paths of length two and pair paths of length three or more. Note that a pair path occurs when there is a *vee* among the semifinal pieces. [5] defined a *vee* to consist of a triple of pieces, one *center* piece and two *base* pieces, with the property that the center piece intersects both of the base two pieces. [5] proved the following lemma that bounds the expected number of vees in the implication graph.

Lemma 3. *If each player only chooses 2 semifinal pieces then the expected number of vees in IG can be as high as $\Theta(n^2)$, which would be disastrous. However, if two brackets of $d = 2$ pieces are chosen and these are narrowed down to two semifinal piece then the expected number of vees in IG is at most $\frac{16d^3}{\alpha^2}n$.*

Using Lemma 3 [5] proved the following lemma that bounds the probability of implication graph having pair paths of length three or more.

Lemma 4. *The probability that the implication graph IG contains a pair path of length at least three is at most $\frac{32d^5}{\alpha^2(\alpha-4d^2)}$.*

A pair path of length two occurs if and only if the implication graph contains a *same-player-vee*. A *same-player-vee* is a vee where both of the base pieces belong to the same player. That is, there is a center piece $a_{\langle p,r\rangle}$ and two bases $a_{\langle q,0\rangle}$ and $a_{\langle q,1\rangle}$. For example, see pieces $a_{\langle 4,0\rangle}$, $a_{\langle 2,0\rangle}$ and $a_{\langle 2,1\rangle}$ in Figure 1. To get around the problem of same-player-vees, they introduced the *same-player-vee graph*. The vertices of the *same-player-vee graph SG* are the n players p, $1 \leq p \leq n$, and if player p and player q are involved in same-player-vee with player p in the center then there is a directed edge from p to q. [5] show how to partition the players into two groups such that there is no same-player-vee involving two players in the same partition. [5] proved Lemma 5 by bounding the probability of same-player-vee graph having a path of length two.

Lemma 5. *The probability that the same-player-vee graph is not $w = 2$ colorable is at most $\frac{16d^3}{\alpha^3} + \frac{8d^2}{\alpha^2}$.*

Finally, because the implication graph on S_h contains no pair paths of any length, the Edmonds-Pruhs protocol ensures that the final piece of at most one player from S_h covers this point. We can then conclude that for any point in the cake, the final pieces of at most $w = 2$ players cover this point.

4 The High Confidence Balanced Allocation Lemma

In this section we give a modification of the Edmonds-Pruhs protocol that will establish the High Confidence Balanced Allocation Lemma for cake. A key concept in the proof of correctness of the Edmonds-Pruhs protocol is the concept of a bad player. A player p is *bad* if a pair path of length three or more starting with p exists in the implication graph, or a path of length two or more starting with p exists in the same-player-vee graph. Edmonds and Pruhs [5] proved that for some constant c, larger than 1, the probability that a particular player is bad is at most $\frac{1}{cn}$. We modify the Edmonds-Pruhs protocol in [5] in the following ways:

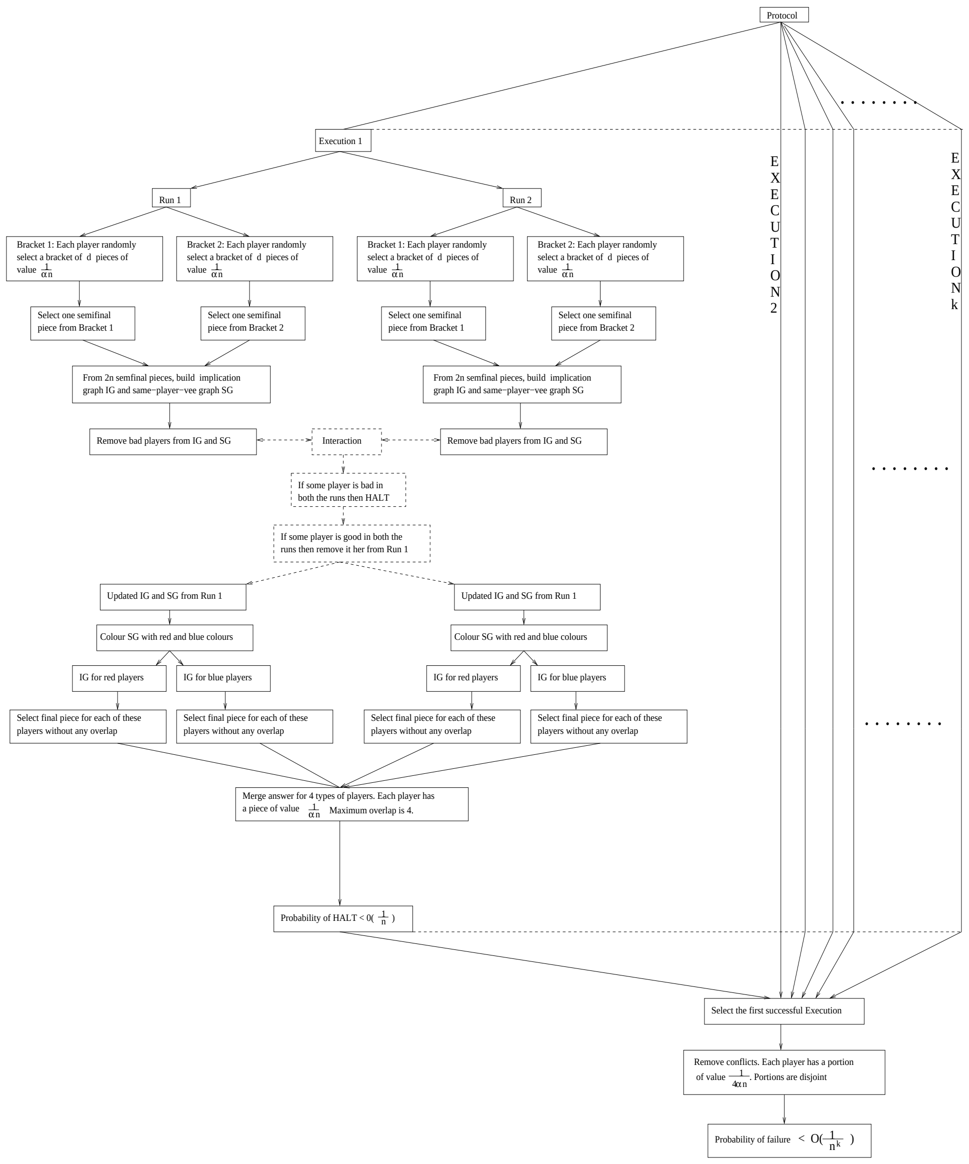
Protocol
Execution 1
Run 1
Run 2
Bracket 1: Each player randomly select a bracket of d pieces of value 1/αn
Bracket 2: Each player randomly select a bracket of d pieces of value 1/αn
Select one semifinal piece from Bracket 1
Select one semifinal piece from Bracket 2
From 2n semfinal pieces, build implication graph IG and same−player−vee graph SG
Remove bad players from IG and SG
Interaction
If some player is bad in both the runs then HALT
If some player is good in both the runs then remove it her from Run 1
Updated IG and SG from Run 1
Colour SG with red and blue colours
IG for red players
IG for blue players
Select final piece for each of these players without any overlap
Merge answer for 4 types of players. Each player has a piece of value 1/αn Maximum overlap is 4.
Probability of HALT < 0(1/n)
EXECUTION 2
EXECUTION k
Select the first successful Execution
Remove conflicts. Each player has a portion of value 1/4αn. Portions are disjoint
Probability of failure < O(1/n^k)

Fig. 2. Flowchart of Our Protocol.

Modified Edmonds-Pruhs Protocol:

- We make two independent runs of the protocol.
- In each run, after the formation of the implication graph and the same-player-vee graph, we remove all bad players. Because of this modification, some players (the ones that are bad in both runs) may not be assigned a candidate piece.

After these two separate runs, some players may be assigned two final pieces, one from each run. In this case, the player need only keep the final piece from the first run. But the key fact is that no point of the cake is covered by more than four candidate pieces, since each run guarantees contention at most two. As before we can then use any proportionally fair protocol to divided the portions of the cake where the final pieces overlap. A flow chart of this protocol is given in Figure 2.

We can then simply calculate the probability that one iteration of this modified protocol fails to assign a final piece to every player:

$$\begin{aligned}
&\mathrm{Prob}[Modified\ Edmonds\ Pruhs\ protocol\ fails] \\
&\leq \mathrm{Prob}[there\ exists\ a\ player\ p\ that\ is\ bad\ in\ both\ runs] \\
&\leq n \times \mathrm{Prob}[player\ p\ is\ bad\ in\ both\ runs] \\
&\leq n \times (\mathrm{Prob}[player\ p\ is\ bad\ in\ one\ run])^2 \\
&\leq n \times O(\tfrac{1}{n})^2 \\
&= O(\tfrac{1}{n})
\end{aligned}$$

Thus the probability that the Modified-Edmonds-Pruhs protocol does not succeed after k applications is at most $O(\frac{1}{n^k})$.

5 Initial Unsuccessful Approach

To us, the most obvious way to modify the Edmonds-Pruhs protocol to obtain a high confidence result was to show that the same-player-vee graph is $O(1)$-colorable with high probability, and to show that the implication graph does not have a pair-path with high probability. We were able to accomplish the former in Lemma 6 by slightly modifying the protocol. But the latter is unfortunately false, there can be pair-paths with high probability.

Lemma 6. *Let $\alpha \geq 10$ be some sufficiently large constant. Assume that each player conceptually partitions the unit interval into αn disjoint candidate subintervals/pieces of equal value. Each player then independently picks $d' = 3d = 6$ of her pieces uniformly at random, with replacement. There is then an efficient method that, with probability at least $(1 - O(\frac{1}{n}))$, chooses three of the d' pieces for each player, and then narrows down two pieces for each player, so that same-player-vee graph build from these chosen pieces can be colored by at most two colors.*

After some reflection, it is clear that a pair path in the implication graph, in and of itself, is not a problem. A pair path from $a_{\langle p,r\rangle}$ to $a_{\langle p,1-r\rangle}$ just implies that we should select $a_{\langle p,1-r\rangle}$ for player p. However, a directed path in both directions, from $a_{\langle p,0\rangle}$ to $a_{\langle p,1\rangle}$ and from $a_{\langle p,1\rangle}$ to $a_{\langle p,0\rangle}$ is problematic since the selection of either piece requires the selection of the other. This leads us to the following definition.

Pair Cycle:A pair cycle in the implication graph is a directed cycle containing both semifinal pieces $a_{\langle p,0\rangle}$ and $a_{\langle p,1\rangle}$ for some player p.

It is not to difficult to see that if the implication graph IG does not contain pair cycle then one can modify the Final Piece Selection Algorithm so that it can select a disjoint final piece for each player. Further, if the edges in the implication graph were independent, using our standard calculations, we find that the probability of a pair-cycle would be $O(\frac{1}{n})$:

$$\mathrm{Prob}[IG\ contain\ a\ pair\ cycle] \approx \sum_{z=2}^{n} \binom{n}{z-1} \left(\frac{1}{\alpha n}\right)^{z} \approx \Theta\left(\frac{1}{n}\right)$$

The reason that this calculation gives $\Theta(1)$ for a pair-path and $\Theta(\frac{1}{n})$ for a pair-cycle is that a pair-cycle has one more edge relative to the number of vertices than does a pair-path. While we know that the edges are not independent, this calculation did give approximately the right probability for a pair-path, and we the saw no reason why this calculation shouldn't also give approximately the right calculation for a pair-cycle. This led us to the following natural conjecture:

Conjecture 1. The probability that for some player p, we have pair paths of length at least three from $a_{\langle p,0\rangle}$ to $a_{\langle p,1\rangle}$ and from $a_{\langle p,1\rangle}$ to $a_{\langle p,0\rangle}$ in the implication graph IG is at most $O(\frac{1}{n})$.

Recall that pair cycle requires that at least one player's both semifinal pieces have to be present in it. In Lemma 7, we were able to prove Conjecture 1 in the case where exactly one player's both semifinal pieces are present in the pair cycle.

Lemma 7. *The probability that for some player p we have pair paths of length at least three from $a_{\langle p,0\rangle}$ to $a_{\langle p,1\rangle}$ and from $a_{\langle p,1\rangle}$ to $a_{\langle p,0\rangle}$ in the implication graph IG and except player p there is no common player involved in both the pair paths, is at most $O(\frac{1}{n})$.*

Our intuition is that the probability of having more than one player repeating in a pair cycle is not so high. But we are unable to prove this. More generally, we wonder whether the selection of the semifinal pieces from the quarterfinal pieces (by throwing out pieces that overlap the most other pieces) leave with a collection of pieces that essentially have the same graph properties as in the balls and bins model.

References

1. Y. AZAR, (1998) On-line Load Balancing, *Online Algorithms - The State of the Art,* Springer, 178-195.
2. Y. AZAR, A. BRODER, A. KARLIN, AND E. UPFAL, (2000) Balanced Allocations, SIAM Journal of Computing 29, 180–200.
3. S.J. BRAMS AND A.D. TAYLOR (1996). *Fair Division – From cake cutting to dispute resolution.* Cambridge University Press, Cambridge.
4. J. EDMONDS AND K. PRUHS, Cake cutting really isn't a piece of cake, *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'2006).*
5. J. EDMONDS AND K. PRUHS, Balanced Allocations of Cake, *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'2006)*, 623-634.
6. S. EVEN AND A. PAZ (1984). A note on cake cutting. *Discrete Applied Mathematics 7*, 285–296.

7. M. Magdon-Ismail, C. Busch, and M.S. Krishnamoorthy (2003). Cake cutting is not a piece of cake. *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS'2003)*, LNCS 2607, Springer Verlag, 596–607.
8. M. Mitzenmacher, A. Richa, and R. Sitaraman 2000 The power of two random choices: A survey of the techniques and results, *Handbook of Randomized Computing*, editors: P. Pardalos, S. Rajasekaran, and J. Rolim, Kluwer.
9. N. Johnson and S. Kotz (1977). Urn Models and Their Application. *John Wiley and Sons*.
10. S.O. Krumke, M. Lipmann, W. de Paepe, D. Poensgen, J. Rambau, L. Stougie, and G.J. Woeginger (2002). How to cut a cake almost fairly. *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'2002)*, 263–264.
11. J.M. Robertson and W.A. Webb (1995). Approximating fair division with a limited number of cuts. *Journal of Combinatorial Theory, Series A 72*, 340–344.
12. J.M. Robertson and W.A. Webb (1998). *Cake-cutting algorithms: Be fair if you can*. A.K. Peters Ltd.
13. J. Sgall and G. J. Woeginger (2003). A lower bound for cake cutting. LNCS 2461, Springer Verlag, 896–901. *Proc. of the 11th Ann. European Symp. on Algorithms (ESA), Lecture Notes in Comput. Sci. 2832*, Springer, 459–469.
14. H. Steinhaus (1948). The problem of fair division. *Econometrica 16*, 101–104.
15. G.J. Woeginger (2002). An approximation scheme for cake division with a linear number of cuts. *Proc. of the 10th Ann. European Symp. on Algorithms (ESA), Lecture Notes in Comput. Sci. 2461*, Springer, 896–901.