Cake Cutting is Not a Piece of Cake

Malik Magdon-Ismail

Costas Busch

M. S. Krishnamoorthy

Rensselaer Polytechnic Institute

N users wish to share a cake

Fair portion : $\frac{1}{N}$ th of cake

The problem is interesting when people have different preferences

Example:

Meg Prefers Yellow Fish

Tom Prefers Cat Fish

Happy

CUT

Happy

Meg's Piece

Tom's Piece

Meg Prefers
Yellow Fish

Tom Prefers
Cat Fish

Unhappy

CUT

Unhappy

Tom's Piece

Meg's Piece

Meg Prefers
Yellow Fish

Tom Prefers
Cat Fish

The cake represents some resource:

- Property which will be shared or divided
- The Bandwidth of a communication line
- Time sharing of a multiprocessor

Fair Cake-Cutting Algorithms:

- Each user gets what she considers to be $1/N^{th}$ of the cake

- Specify how each user cuts the cake

- The algorithm doesn't need to know the user's preferences

For N users it is known how to divide the cake fairly with $O(N \log N)$ cuts

Steinhaus 1948: "The problem of fair division"

It is not known if we can do better than $O(N \log N)$ cuts

Our contribution:

We show that $\Omega(N \log N)$ cuts are required for the following classes of algorithms:

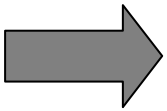
- Phased Algorithms (many algorithms)
- Labeled Algorithms (all known algorithms)

Our contribution:

We show that $\Omega(N^2)$ cuts are required for special cases of envy-free algorithms:

Each user feels she gets more than the other users

Talk Outline



Cake Cutting Algorithms

Lower Bound for Phased Algorithms

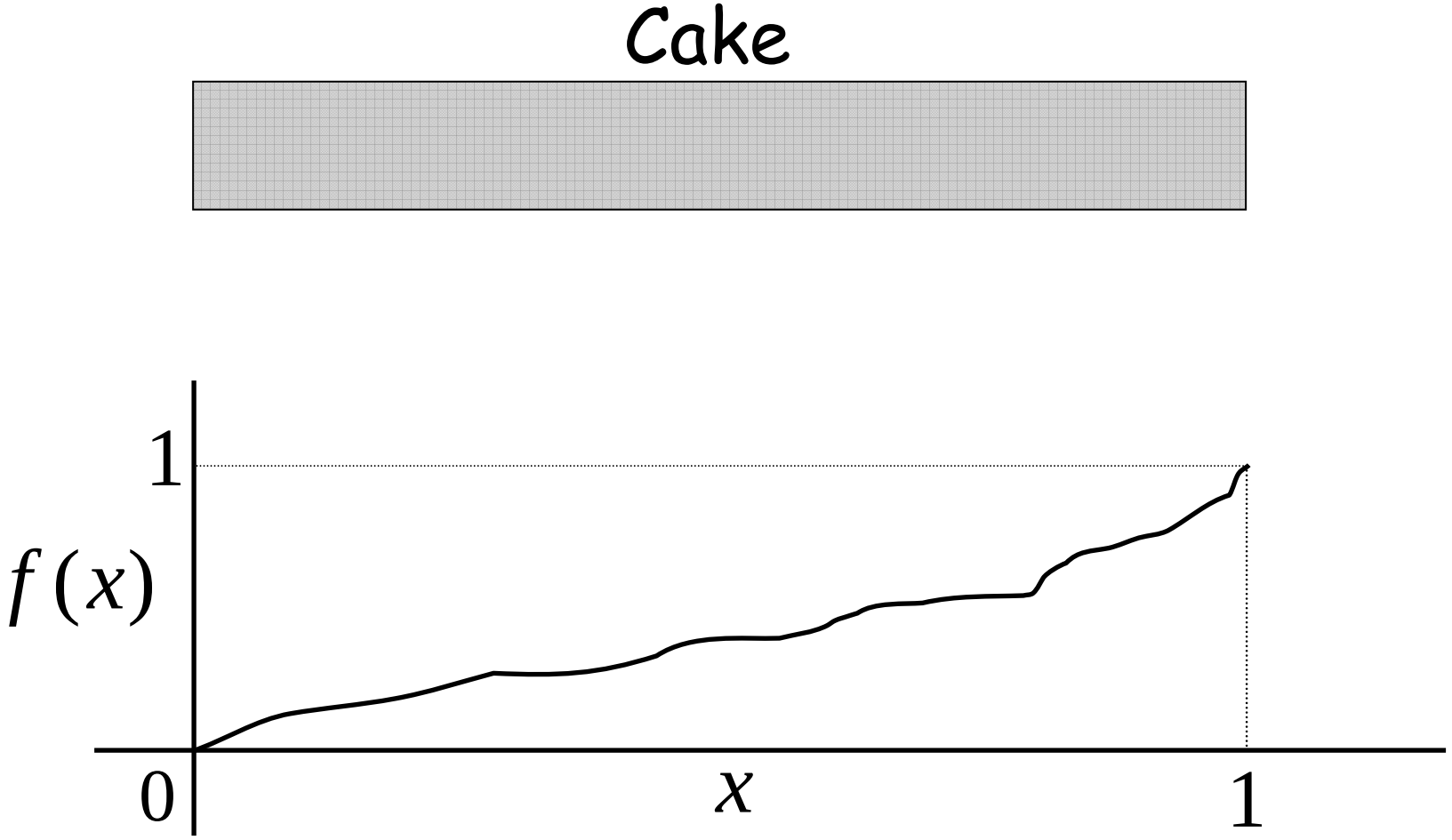
Lower Bound for Labeled Algorithms

Lower Bound for Envy-Free Algorithms

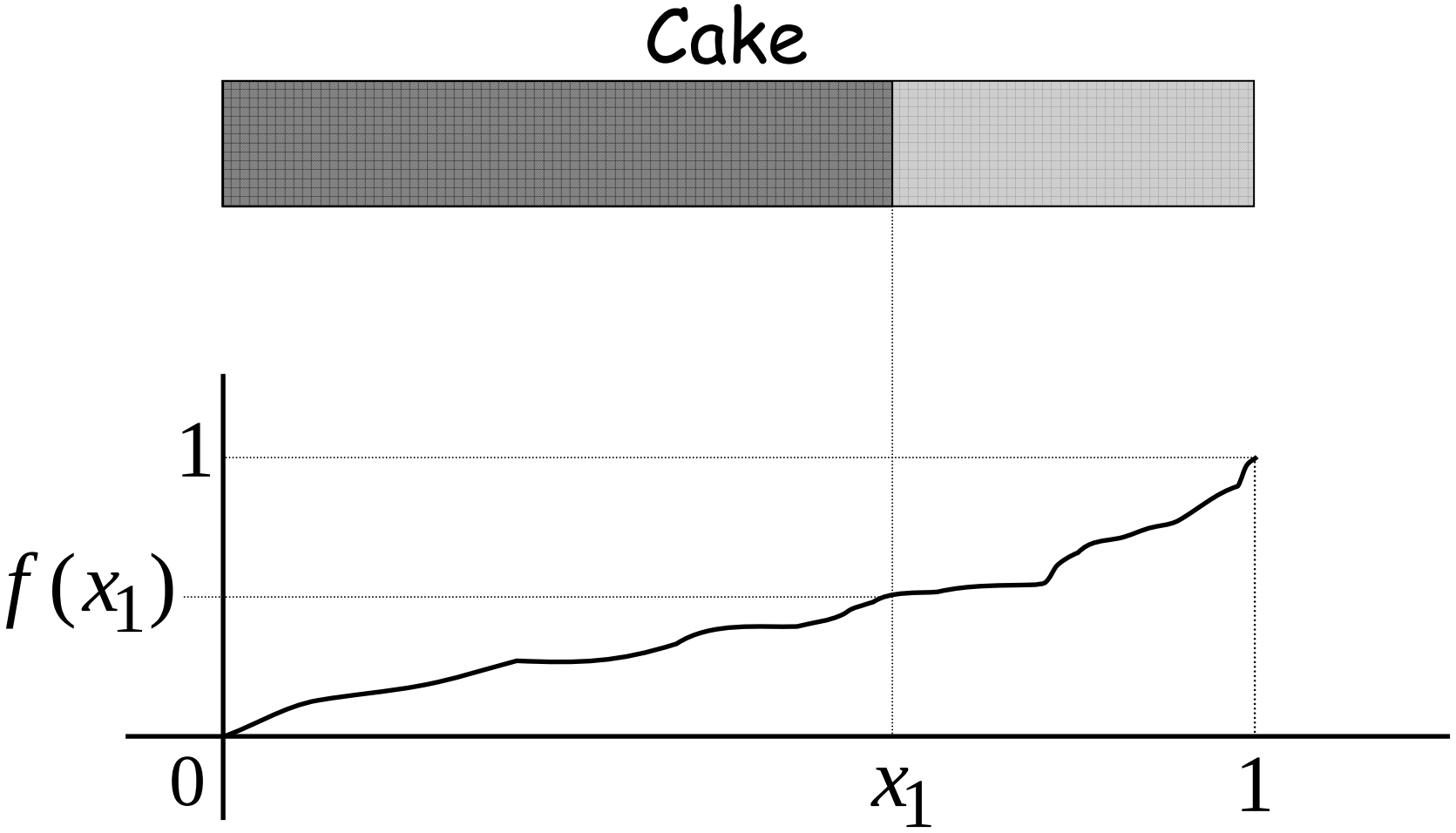
Conclusions


Cake
knife

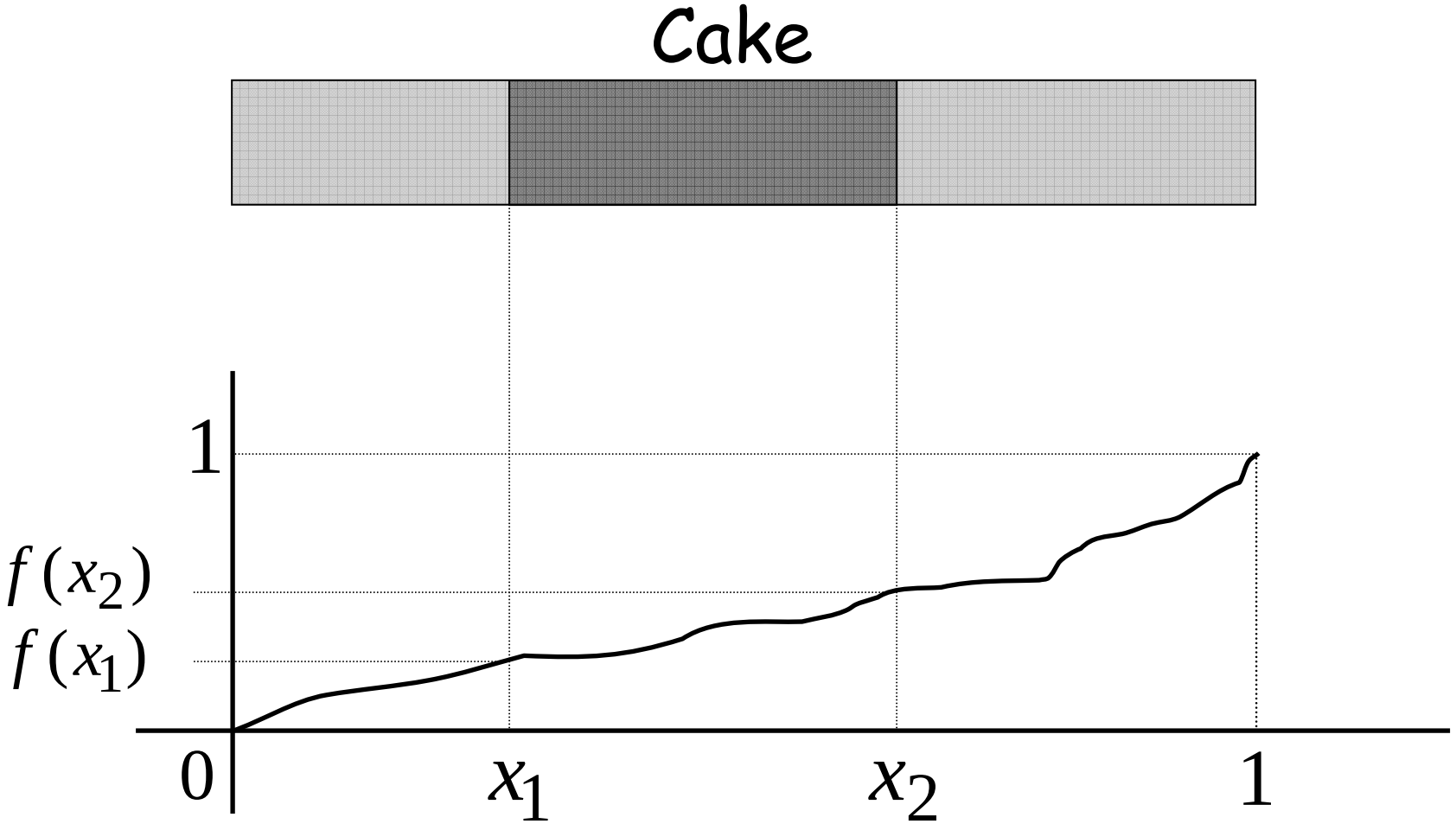

Cake
knife
cut



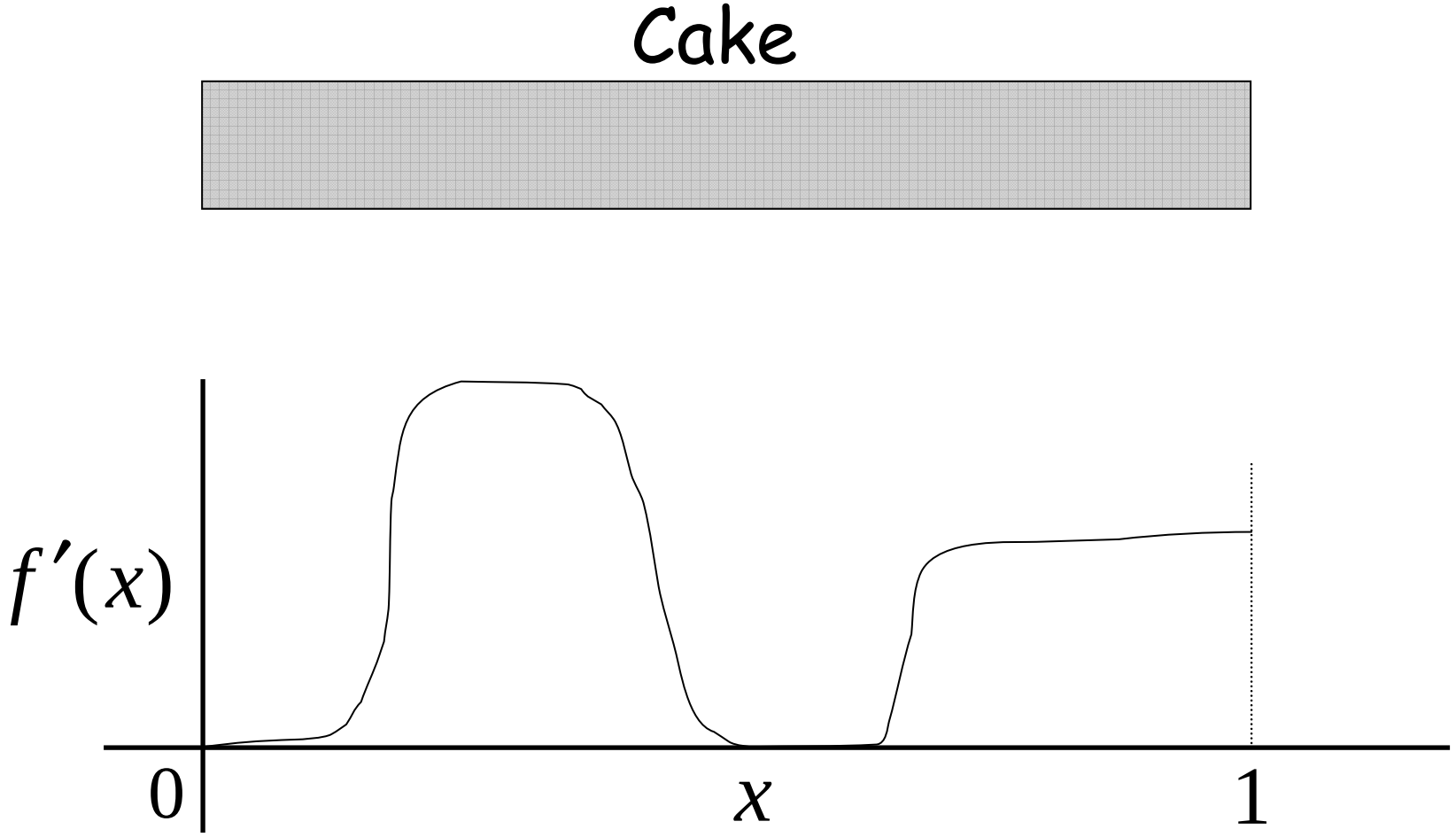
Utility Function for user u_i



Value of piece: $f(x_1)$

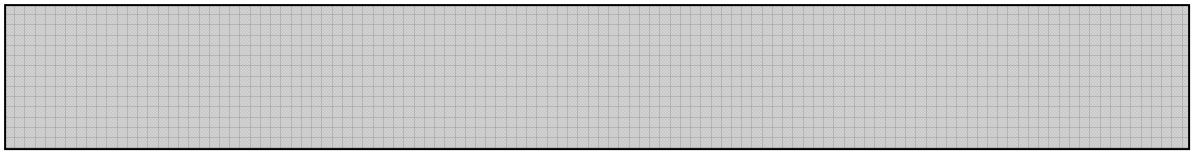


Value of piece: $f(x_2) - f(x_1)$



Utility Density Function for user u_i

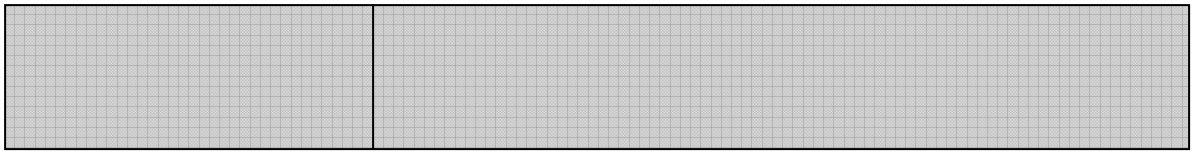
"I cut you choose"



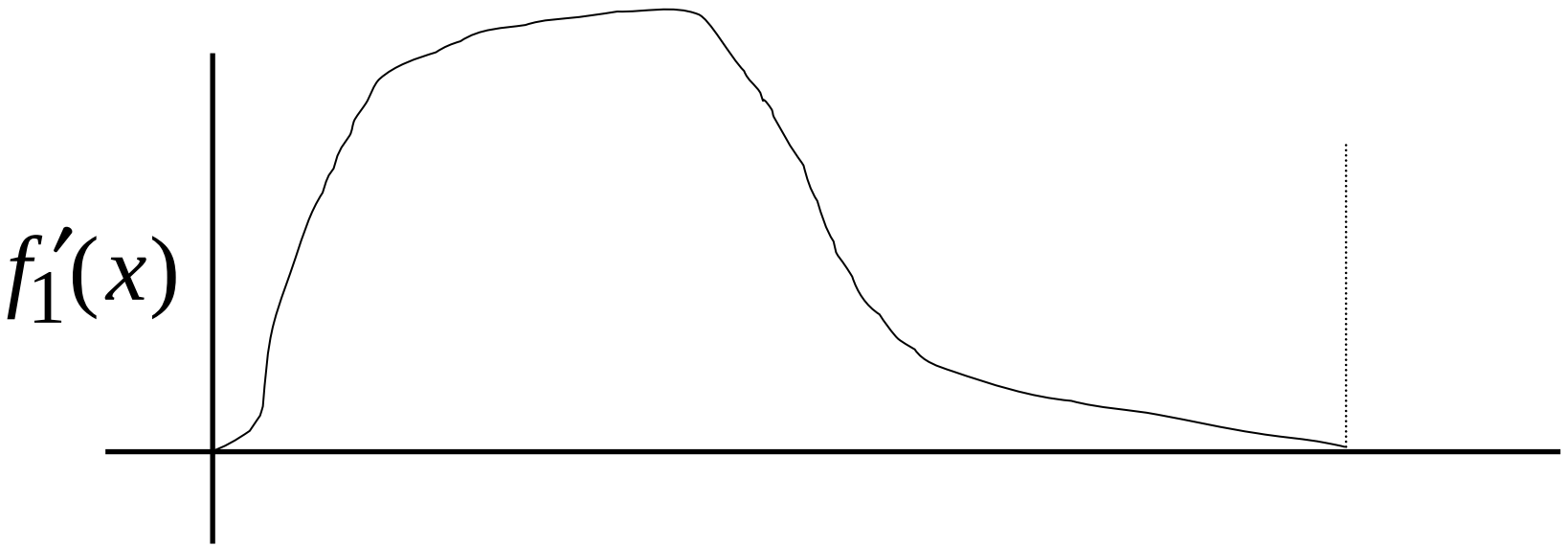
Step 1: User 1 cuts at $1/2$

Step 2: User 2 chooses a piece

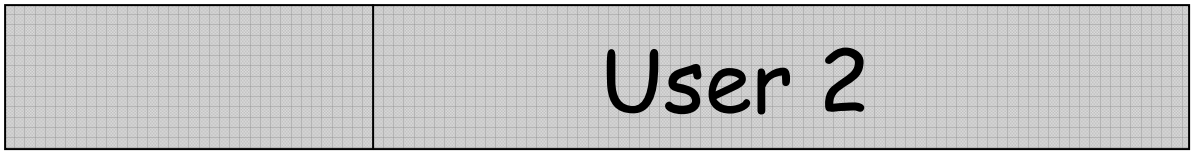
"I cut you choose"



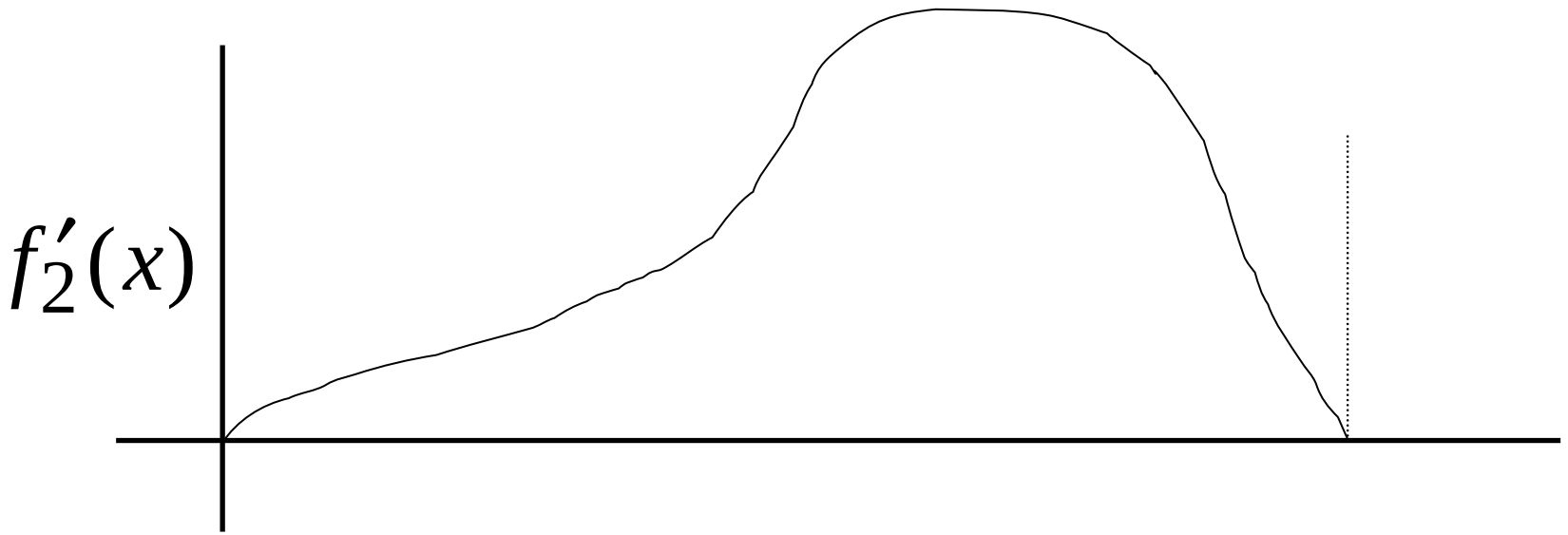
Step 1: User 1 cuts at $1/2$



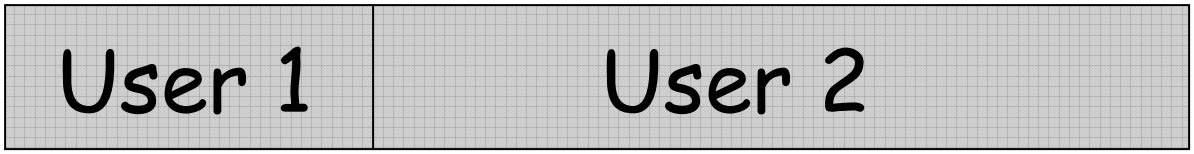
"I cut you choose"



Step 2: User 2 chooses a piece



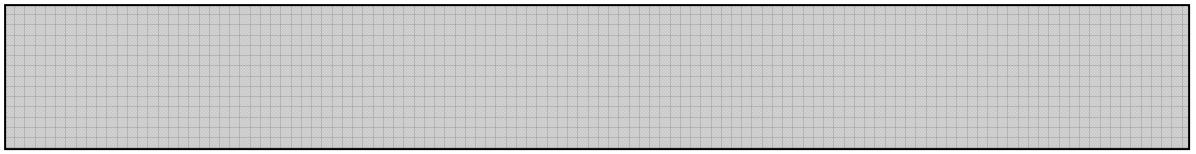
"I cut you choose"



Both users get at least $1/2$ of the cake

Both are happy

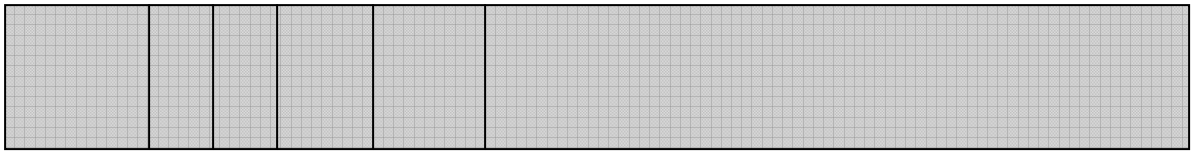
Algorithm A



N users

Phase 1: Each user cuts at $\frac{1}{N}$

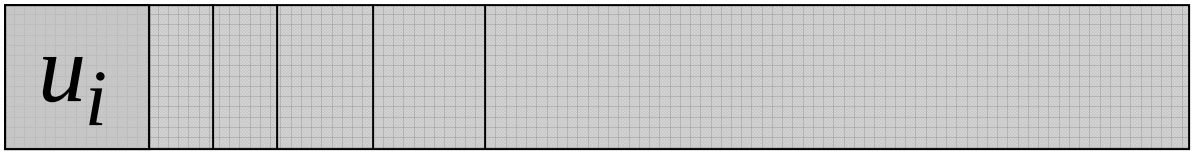
Algorithm A



N users

Phase 1: Each user cuts at $\frac{1}{N}$

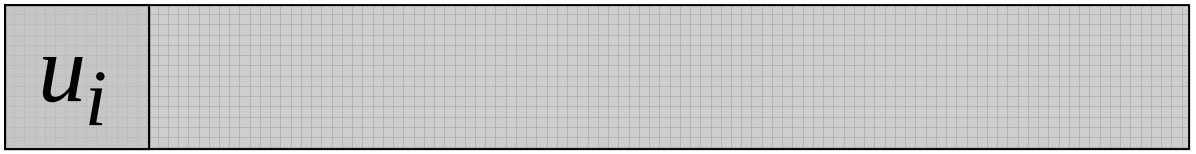
Algorithm A



N users

Phase 1: Give the leftmost piece to the respective user

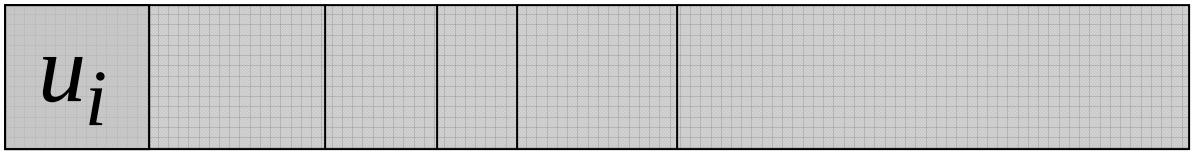
Algorithm A



$N-1$ users

Phase 2: Each user cuts at $\frac{1}{N-1}$

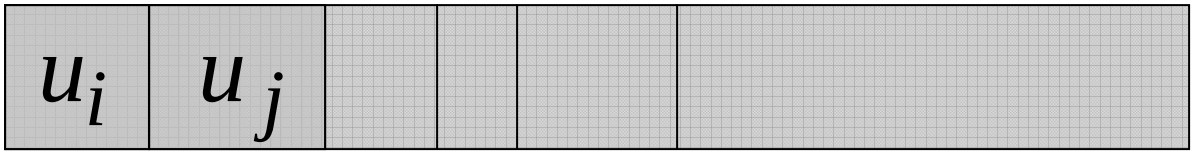
Algorithm A



$N-1$ users

Phase 2: Each user cuts at $\frac{1}{N-1}$

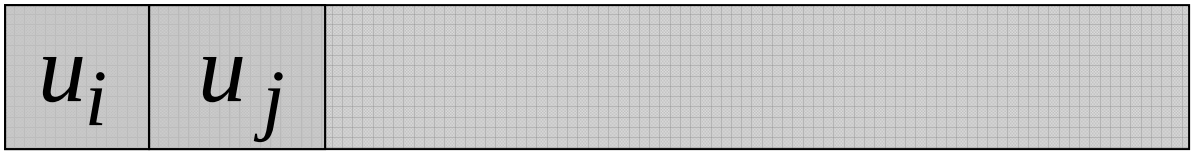
Algorithm A



$N-1$ users

Phase 2: Give the leftmost piece to the respective user

Algorithm A

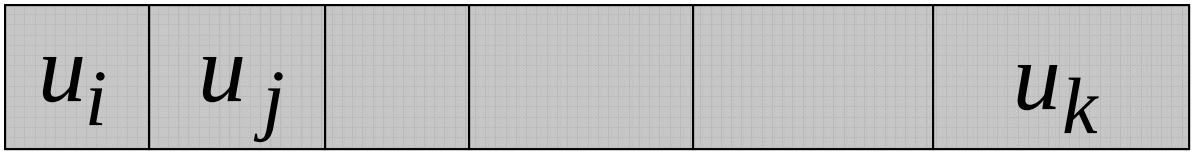


$N-2$ users

Phase 3: Each user cuts at $\frac{1}{N-2}$

And so on…

Algorithm A

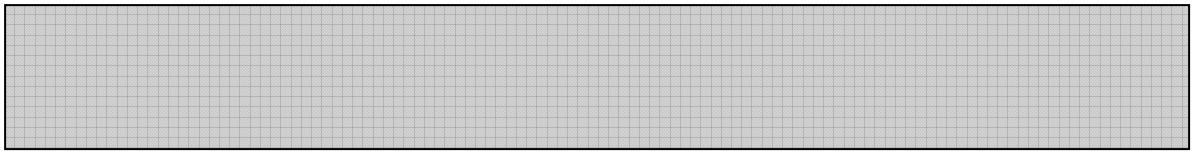


Total number of phases: $N-1$

Total number of cuts:

$$N+(N-1)+(N-2)+\cdots+1 \;=\; O(N^2)$$

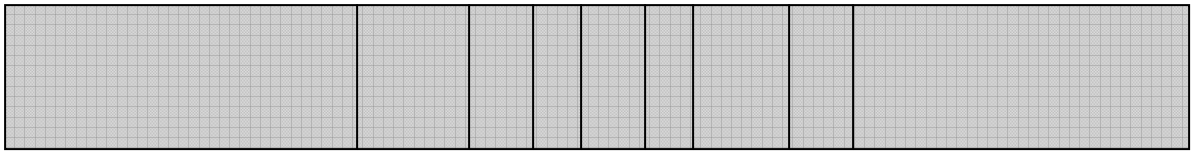
Algorithm B



N users

Phase 1: Each user cuts at $\frac{1}{2}$

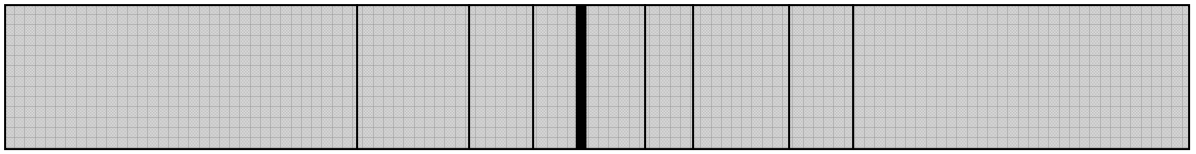
Algorithm B



N users

Phase 1: Each user cuts at $\frac{1}{2}$

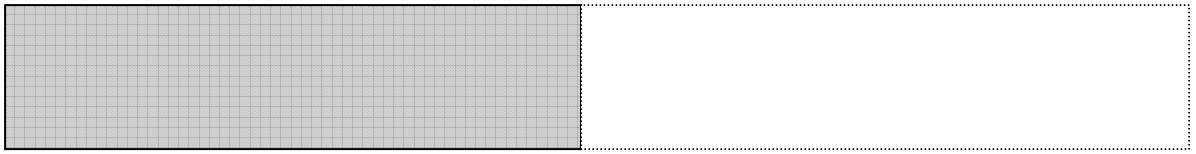
Algorithm B



$\frac{N}{2}$ users $\frac{N}{2}$ users

Phase 1: Find middle cut

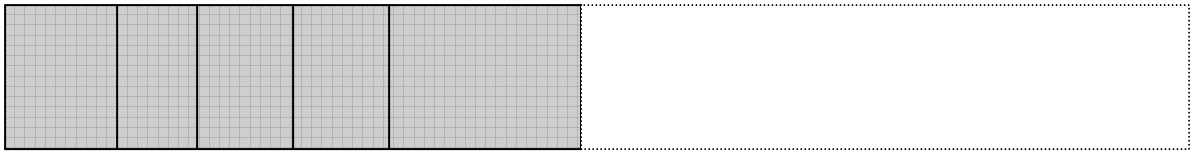
Algorithm B



$\frac{N}{2}$ users

Phase 2: Each user cuts at $\frac{1}{2}$

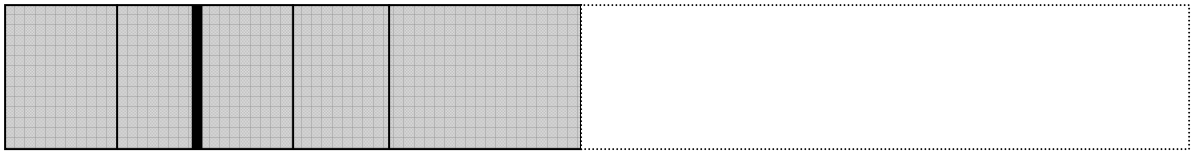
Algorithm B



$\frac{N}{2}$ users

Phase 2: Each user cuts at $\frac{1}{2}$

Algorithm B



$\frac{N}{4}$ $\frac{N}{4}$ users

Phase 2: Find middle cut

Algorithm B

$\frac{N}{4}$ users

Phase 3: Each user cuts at $\frac{1}{2}$

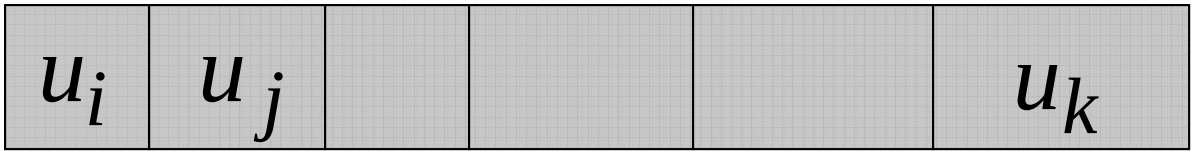
And so on…

Algorithm B

1 user

Phase log N: The user is assigned the piece

Algorithm B



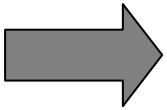
Total number of phases: $\log N$

Total number of cuts:

$$\underbrace{N + N + N + \cdots + N}_{\log N} = O(N \log N)$$

Talk Outline

Cake Cutting Algorithms



Lower Bound for Phased Algorithms

Lower Bound for Labeled Algorithms

Lower Bound for Envy-Free Algorithms

Conclusions

Phased algorithm: consists of a sequence of phases

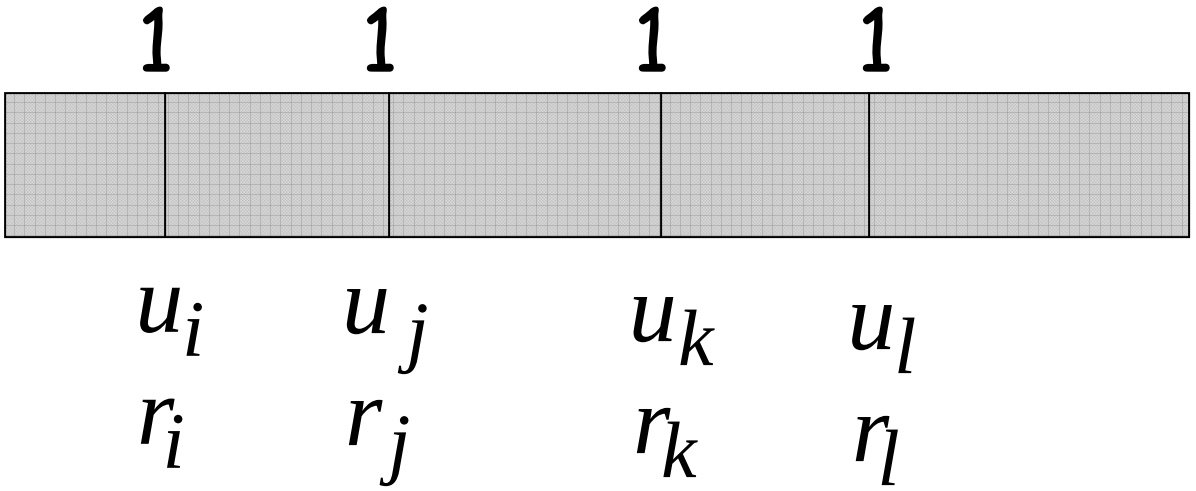
At each phase:

Each user cuts a piece which is defined in previous phases

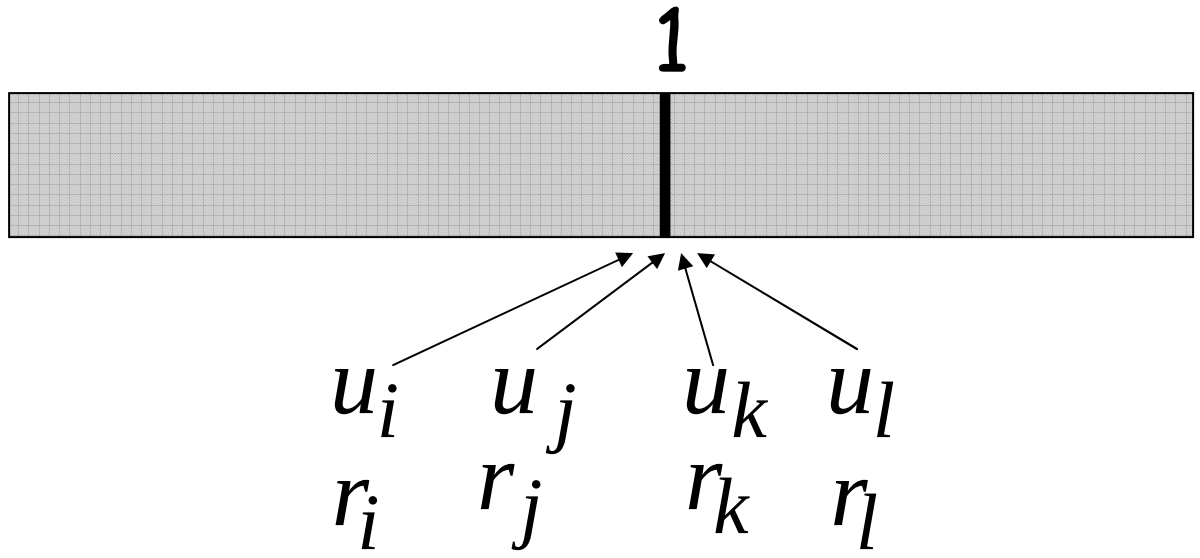
A user may be assigned a piece in any phase

Observation: Algorithms A and B are phased

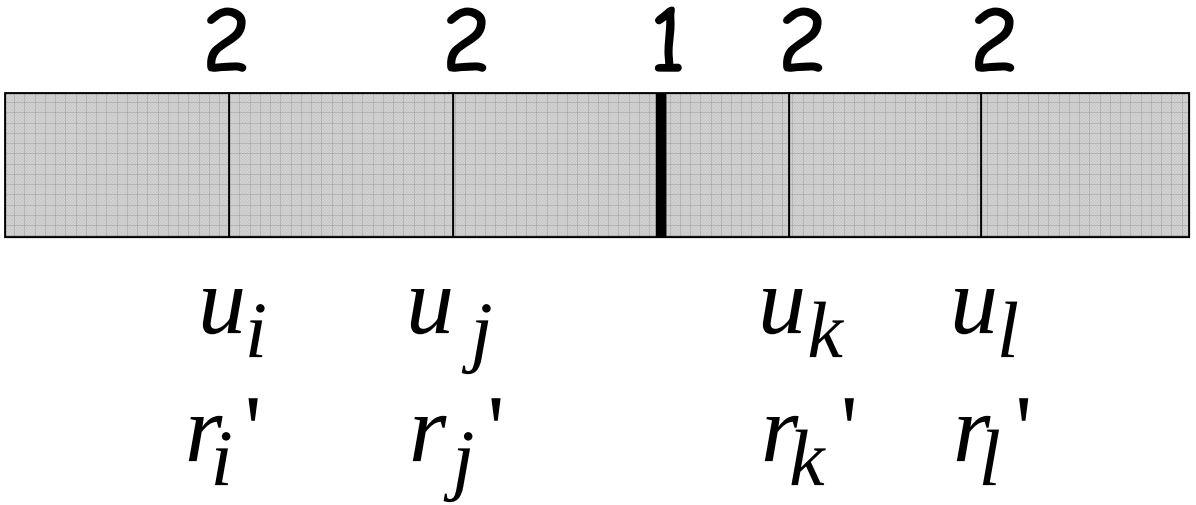
We show: $\Omega(N \log N)$ cuts are required to assign positive valued pieces



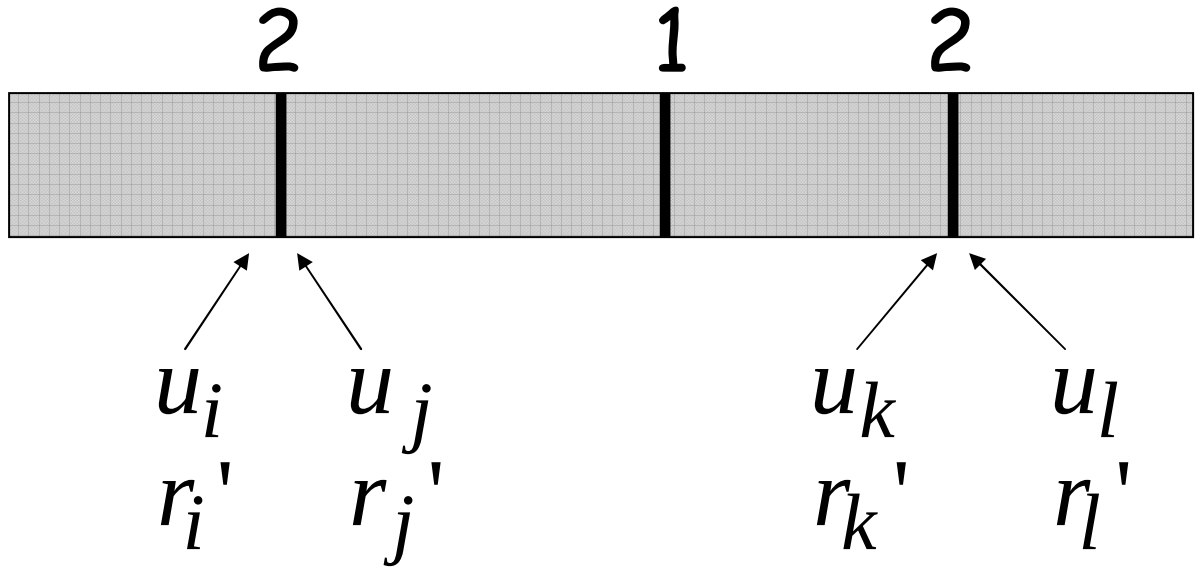
Phase 1: Each user cuts according to some ratio



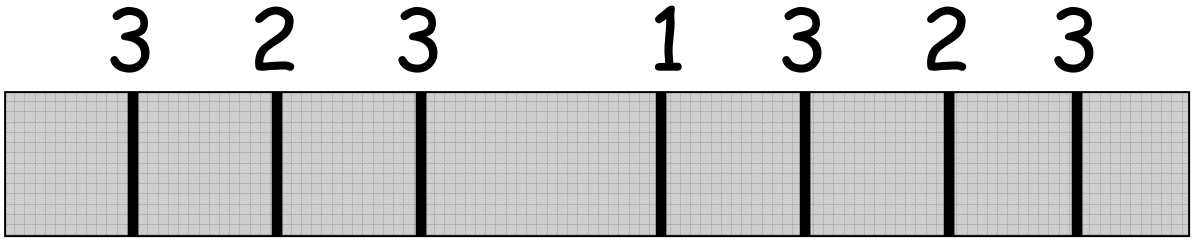
There exist utility functions such that the cuts overlap



Phase 2: Each user cuts according to some ratio

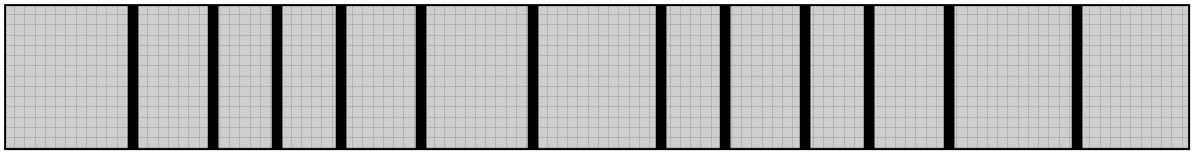


There exist utility functions
such that the cuts in each piece overlap



Phase 3: number of pieces
at most are doubled

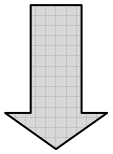
And so on…



Phase k: Number of pieces at most 2^k

For N users:

we need at least N pieces



we need at least $\log N$ phases

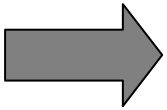
Phase	Users (min)	Pieces (max)	Cuts (min)
1	N	2	N
2	$N-2$	4	$N-2$
3	$N-4$	8	$N-4$
.....	.....	.....	.....
$\log N+1$	0	$2N$	0

Total Cuts: $\Omega(N \log N)$

Talk Outline

Cake Cutting Algorithms

Lower Bound for Phased Algorithms

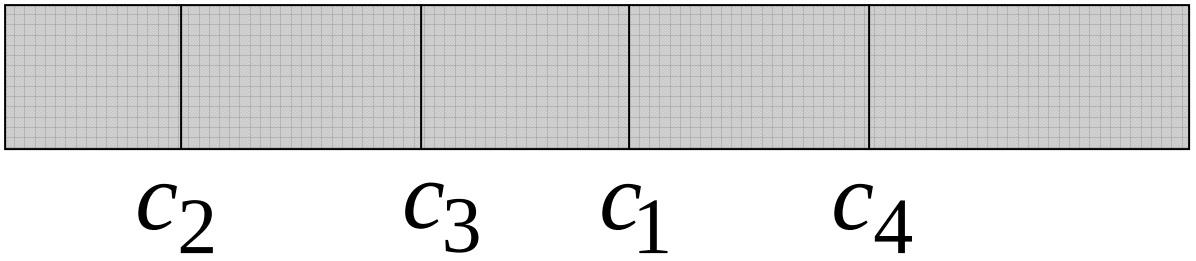


Lower Bound for Labeled Algorithms

Lower Bound for Envy-Free Algorithms

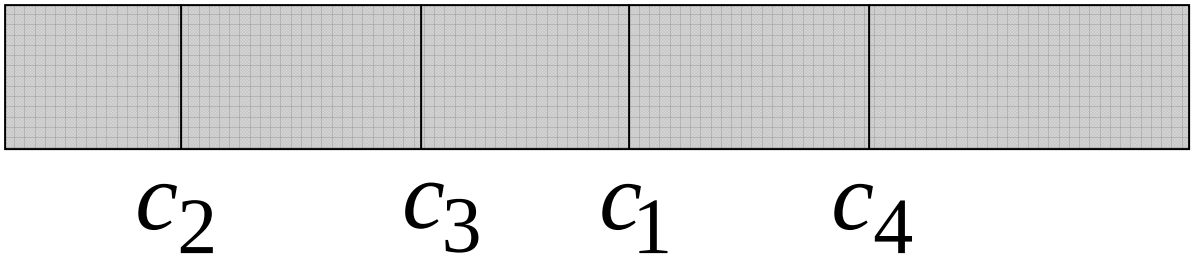
Conclusions

Labels: 00 010 011 10 11

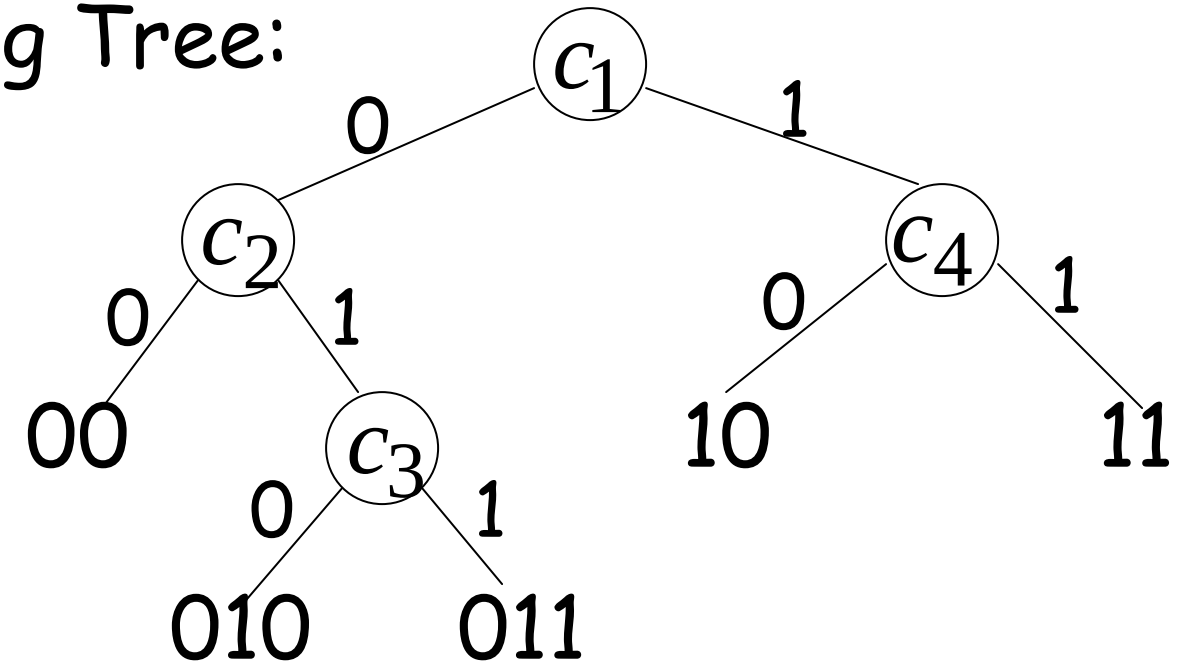


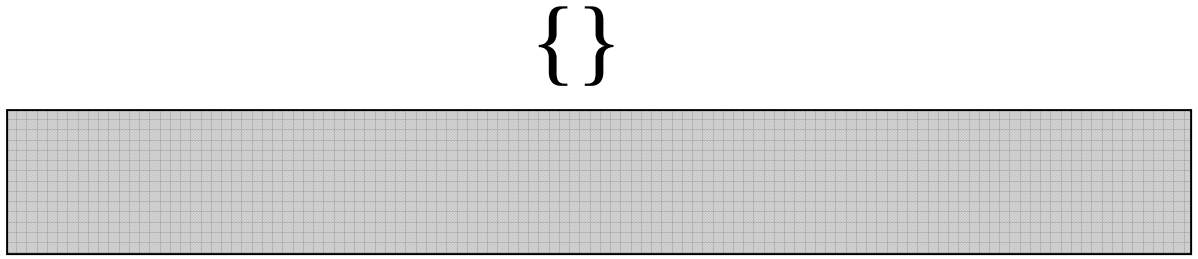
Labeled algorithms: each piece has a label

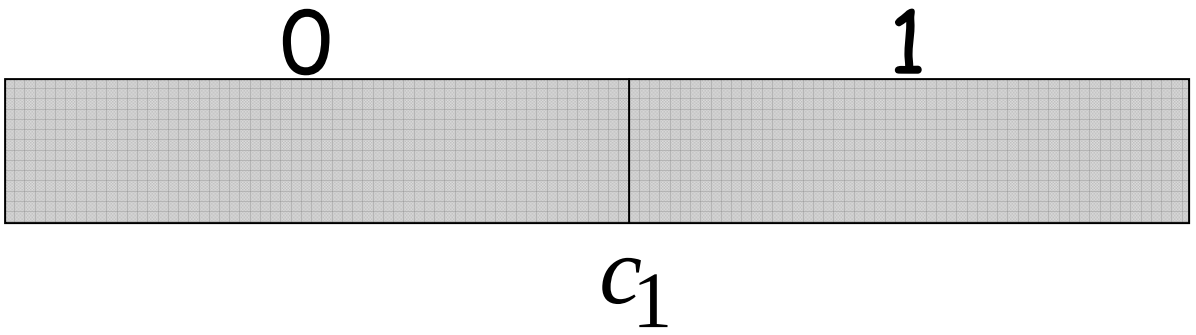
Labels: 00 010 011 10 11

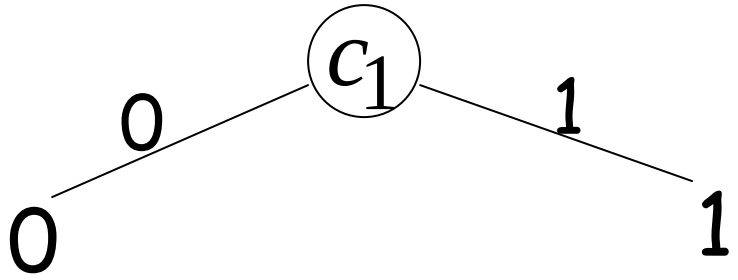


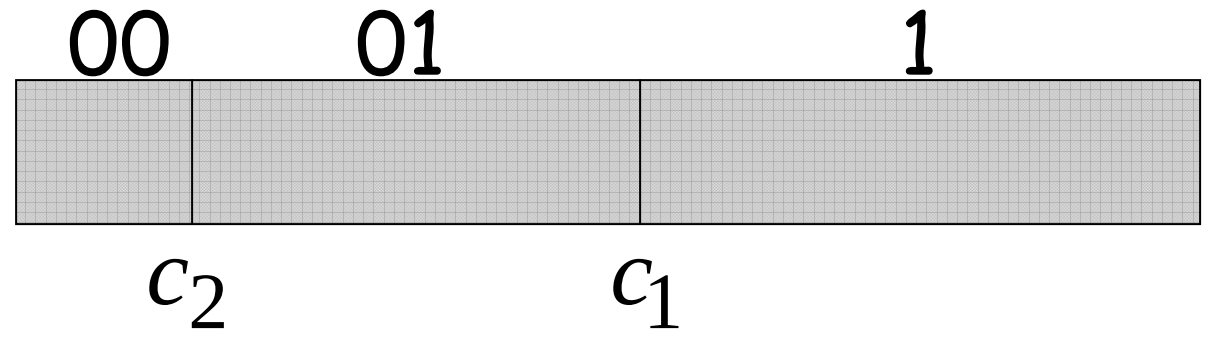
Labeling Tree:

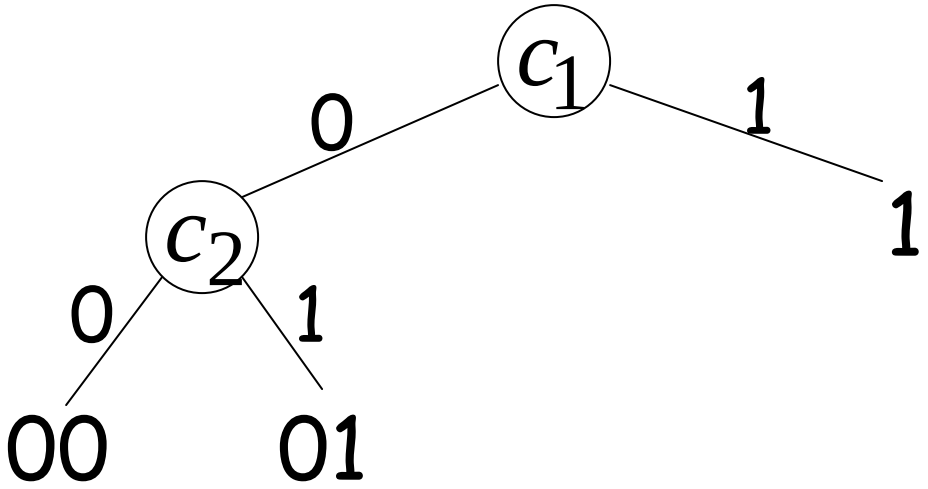


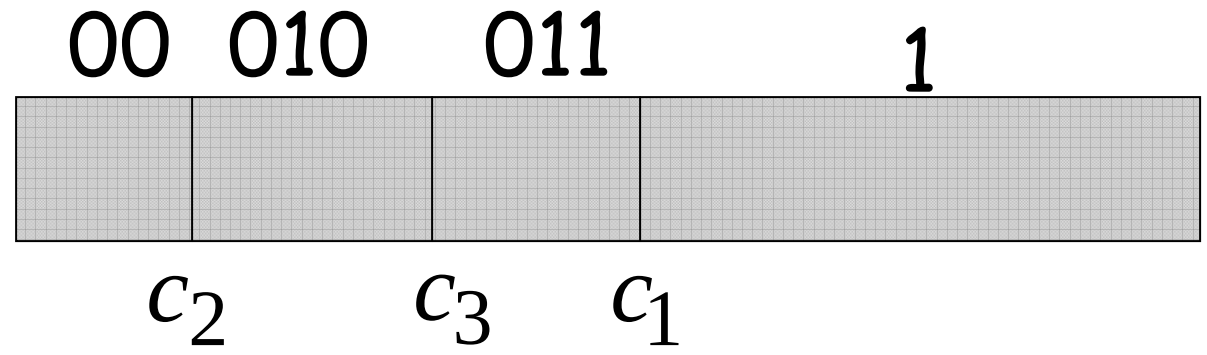


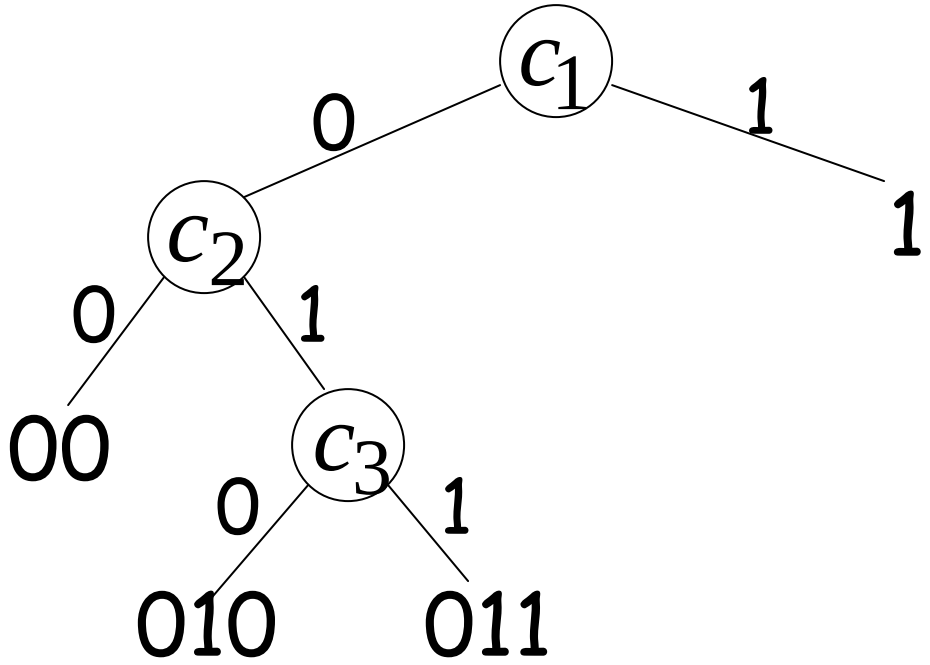

0
1
c_1

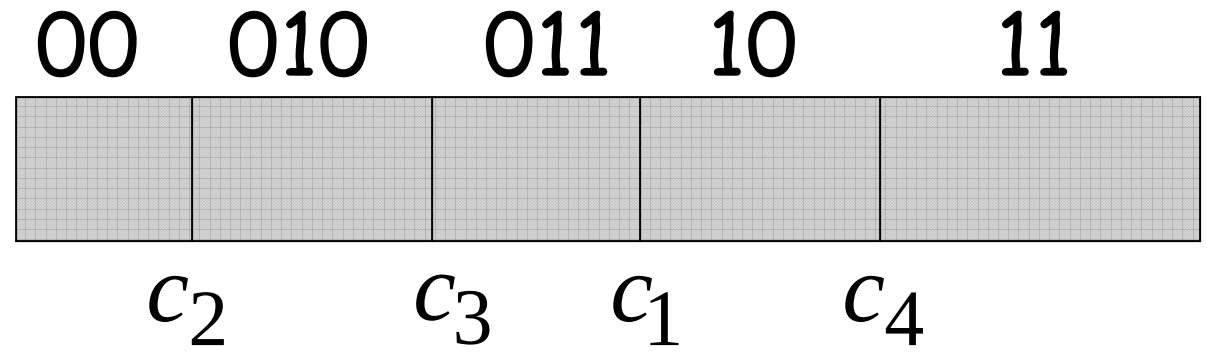

c_1
0
1
0
1

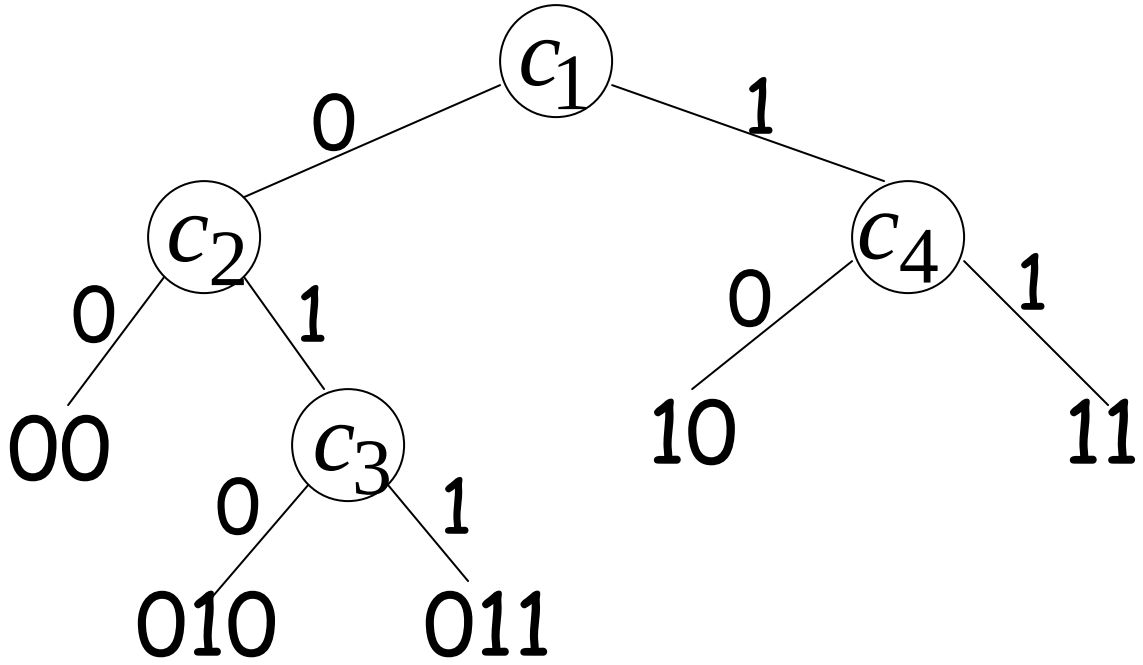

00
01
1
c_2
c_1


c_1
0
1
c_2
1
0
1
00
01

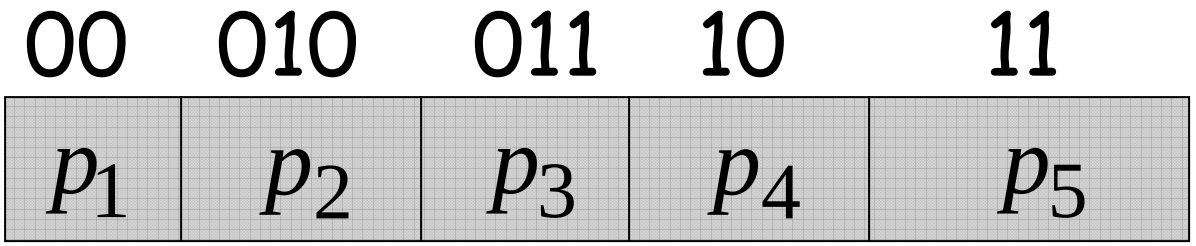

00
010
011
1
c_2
c_3
c_1


c_1
0
1
1
c_2
0
1
00
c_3
0
1
010
011


00
010
011
10
11
c_2
c_3
c_1
c_4


c_1
0
1
c_2
c_4
0
1
0
1
00
c_3
10
11
0
1
010
011

Sorting Labels



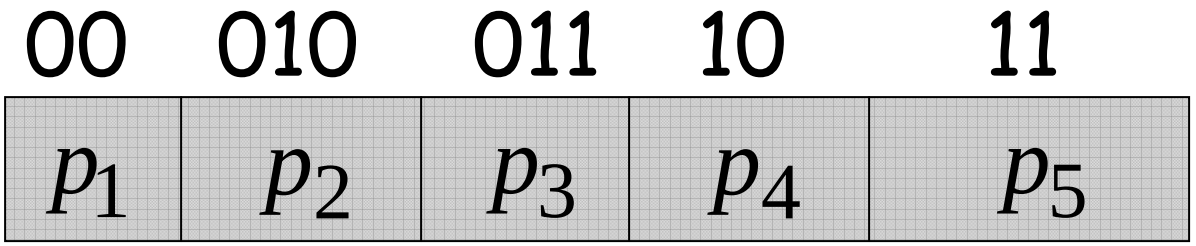
Users receive pieces in arbitrary order:

p_3 p_2 p_5 p_1 p_4

We would like to sort the pieces:

p_1 p_2 p_3 p_4 p_5

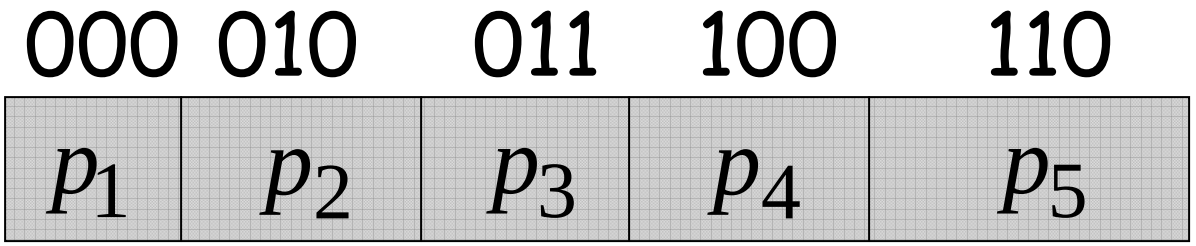
Sorting Labels



p_3 p_2 p_5 p_1 p_4

Labels will help to sort the pieces

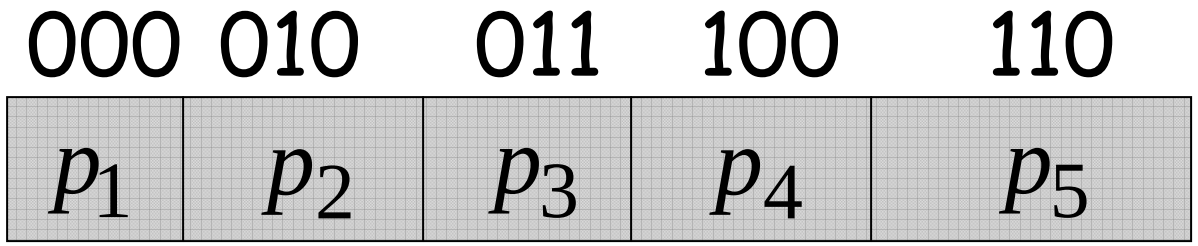
Sorting Labels



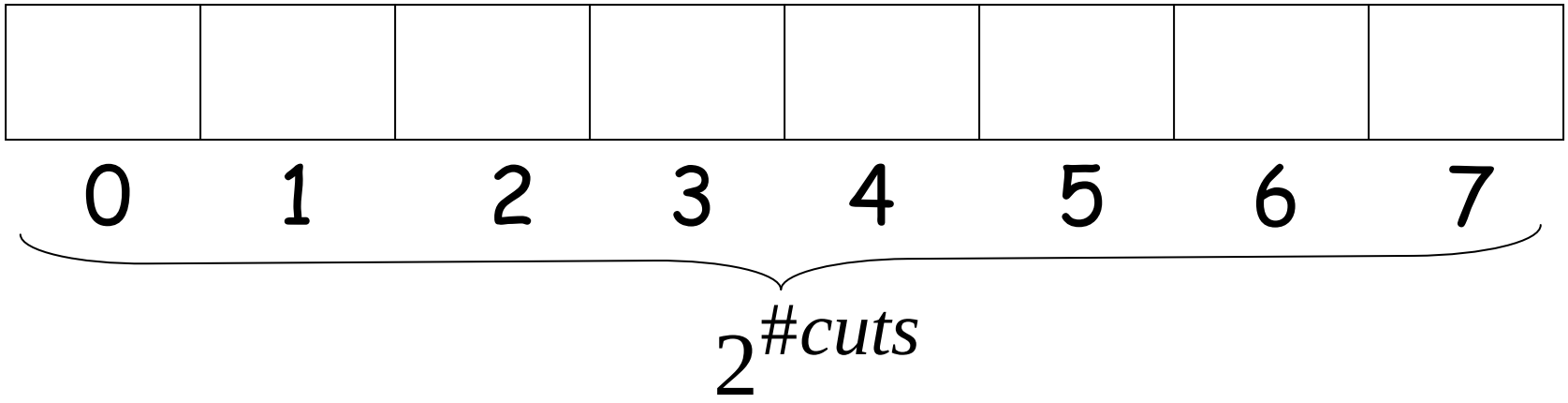
p_3 p_2 p_5 p_1 p_4

Normalize the labels

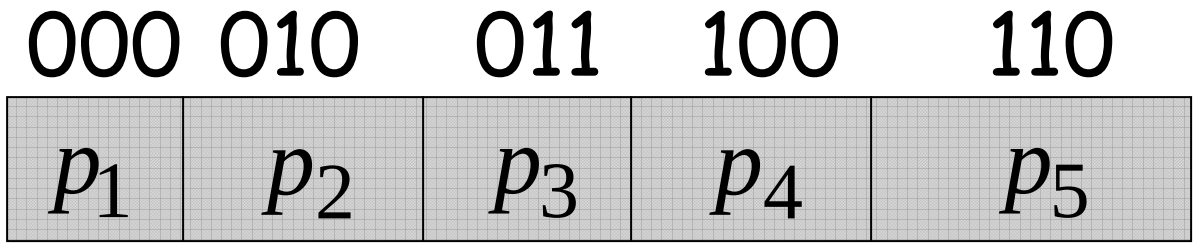
Sorting Labels



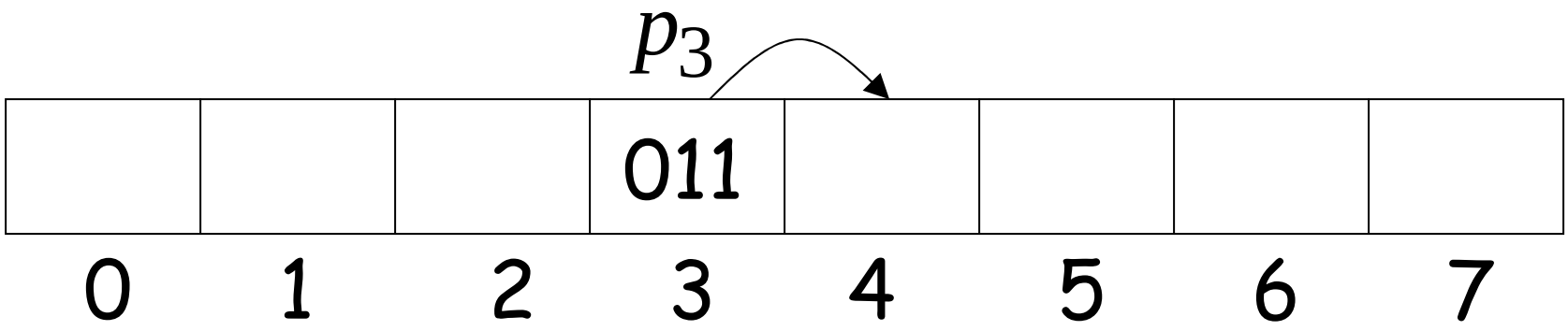
p_3 p_2 p_5 p_1 p_4



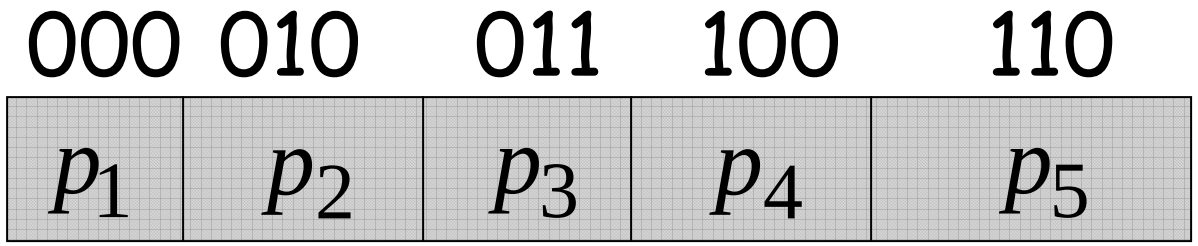
Sorting Labels



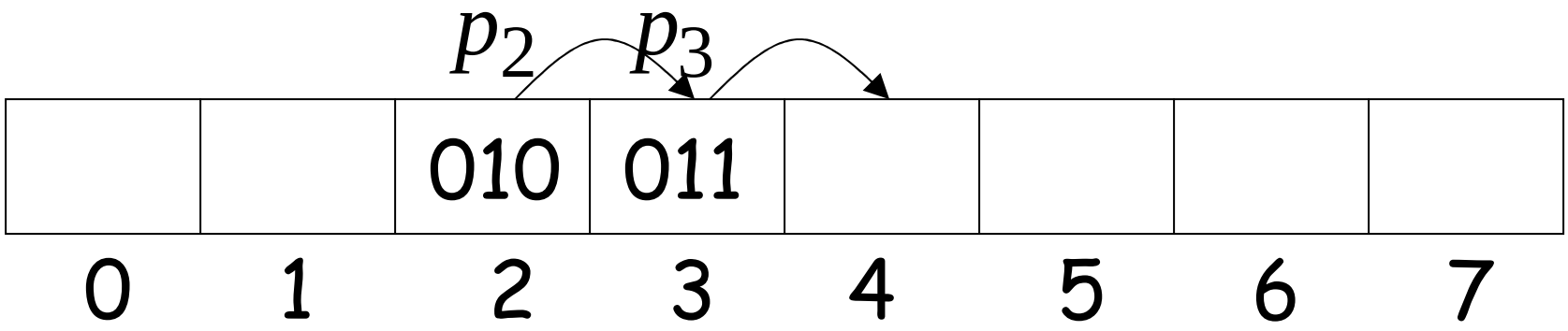
p_2 p_5 p_1 p_4



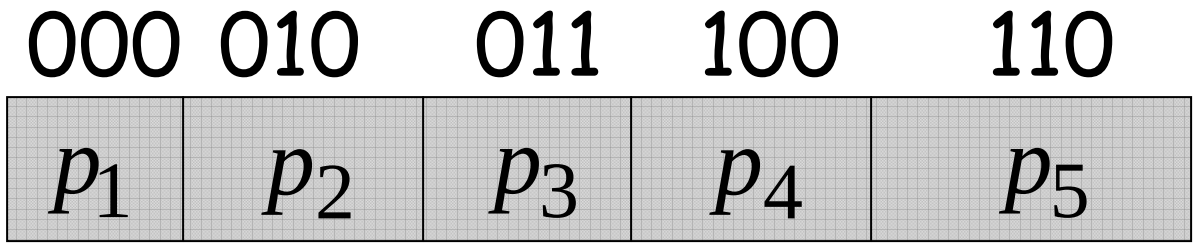
Sorting Labels



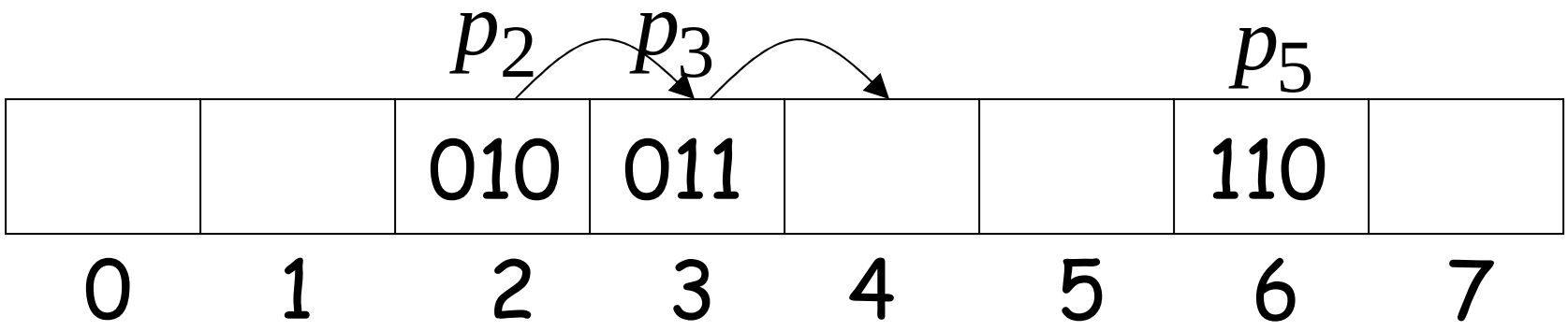
p_5 p_1 p_4



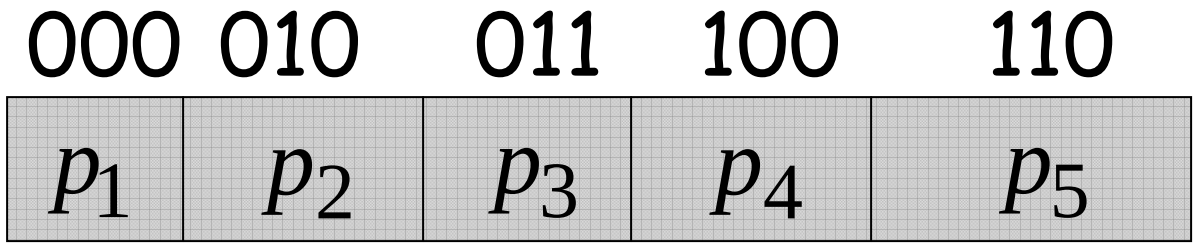
Sorting Labels



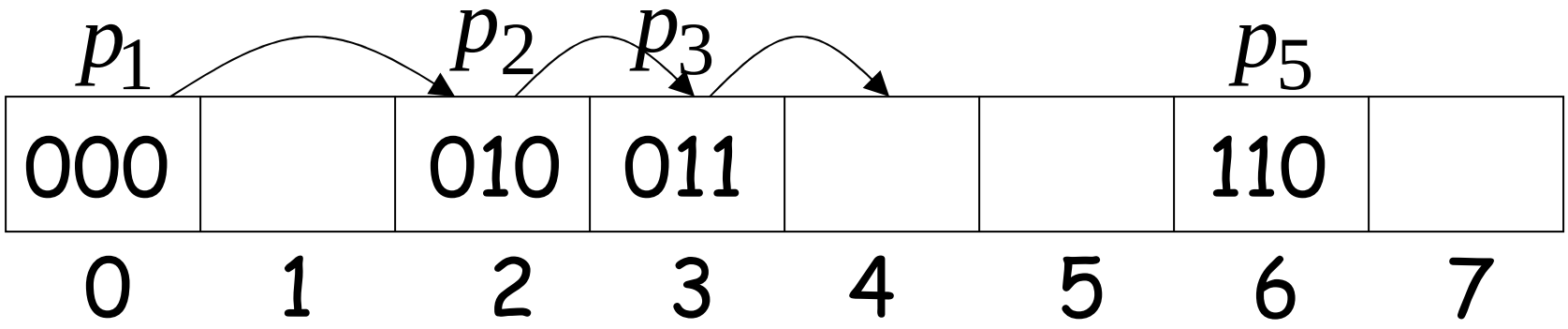
p_1 p_4



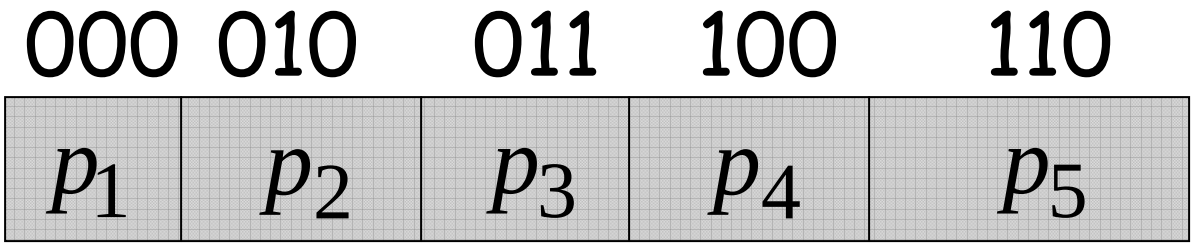
Sorting Labels



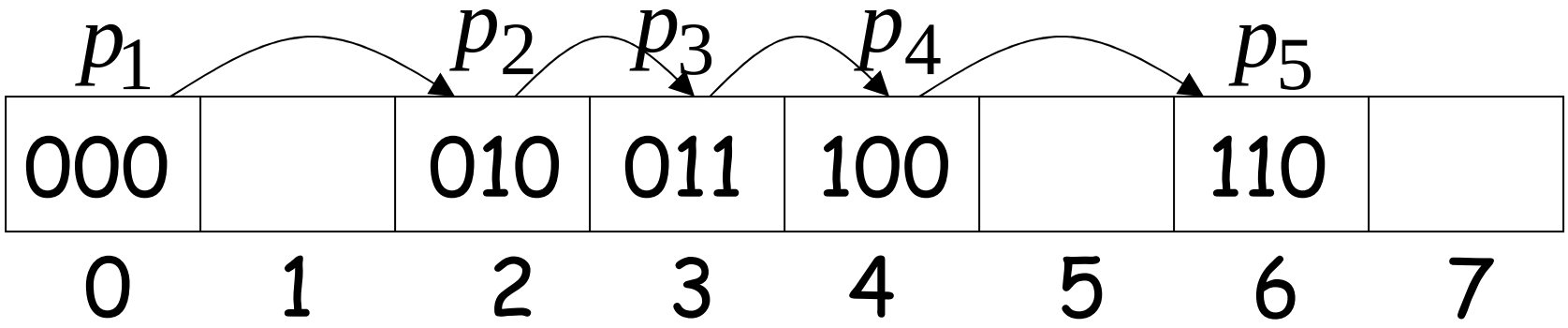
p_4



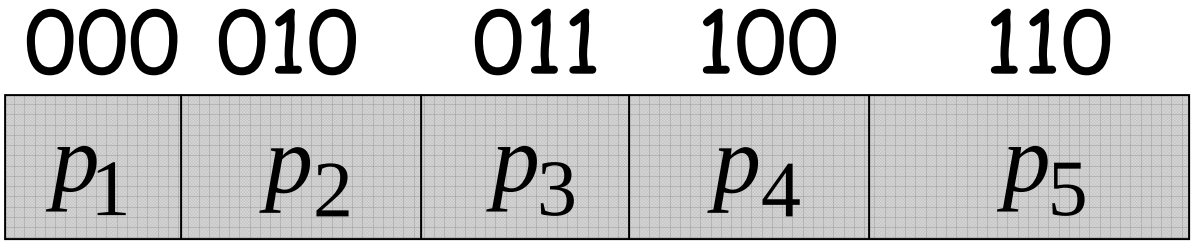
Sorting Labels



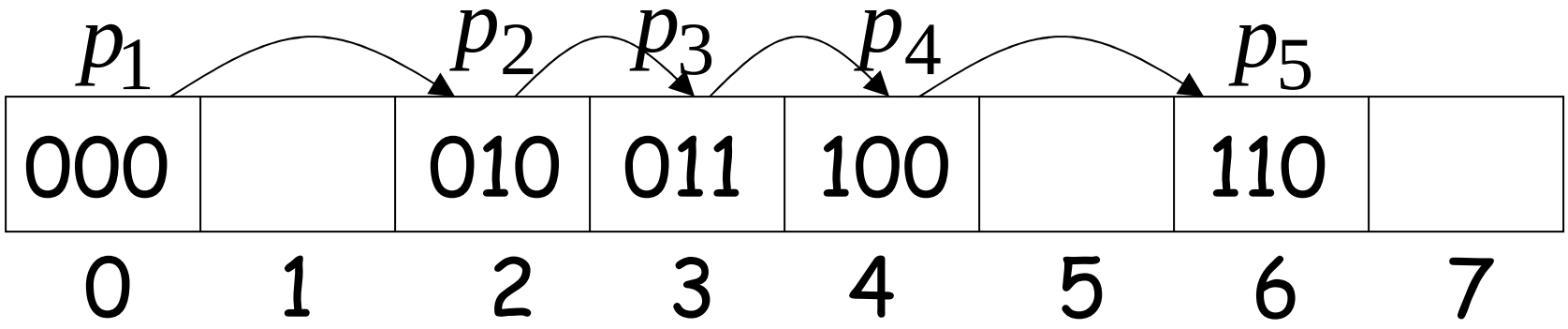
Labels and pieces are ordered!



Sorting Labels



Time needed: $O(\#cuts)$



linearly-labeled &
comparison-bounded algorithms:

Require $O(\#cuts)$ comparisons

(including handling and sorting labels)

Observation: Algorithms A and B are linearly-labeled & comparison-bounded

Conjecture: <u>All</u> known algorithms are linearly-labeled & comparison-bounded

We will show that $\Omega(N \log N)$ cuts are needed for linearly-labeled & comparison-bounded algorithms

Reduction of Sorting to Cake Cutting

Input:

N distinct positive integers: $x_1, \quad x_2, \quad \ldots, \quad x_N$

Output:

Sorted order: $x_k > x_j > \cdots > x_i$

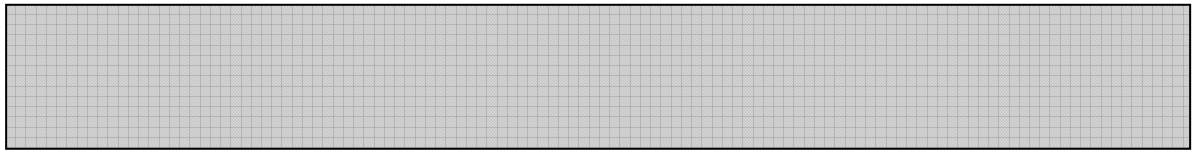
Using a cake-cutting algorithm

N distinct positive integers: $x_1, \quad x_2, \quad \ldots, \quad x_N$

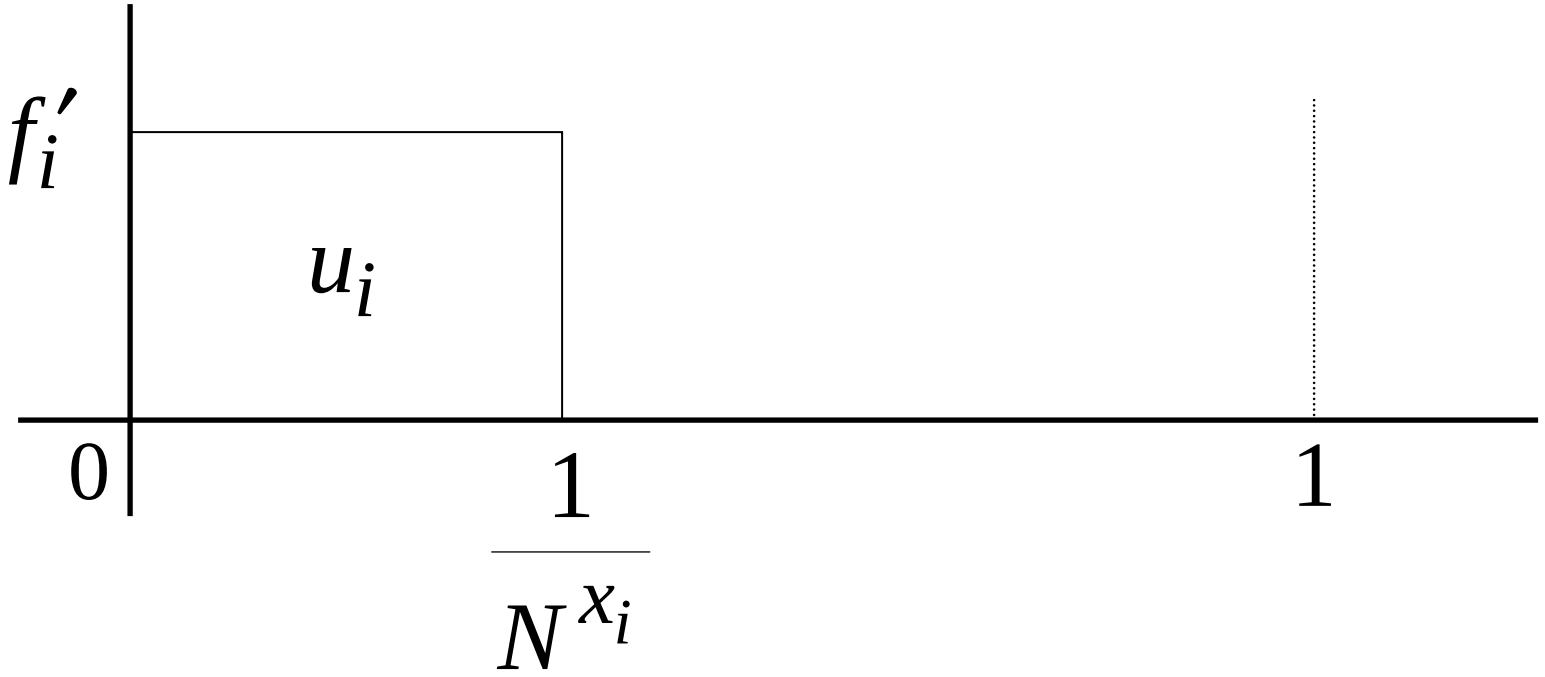
N utility functions: $\quad f_1 \quad f_2 \quad \cdots \quad f_N$

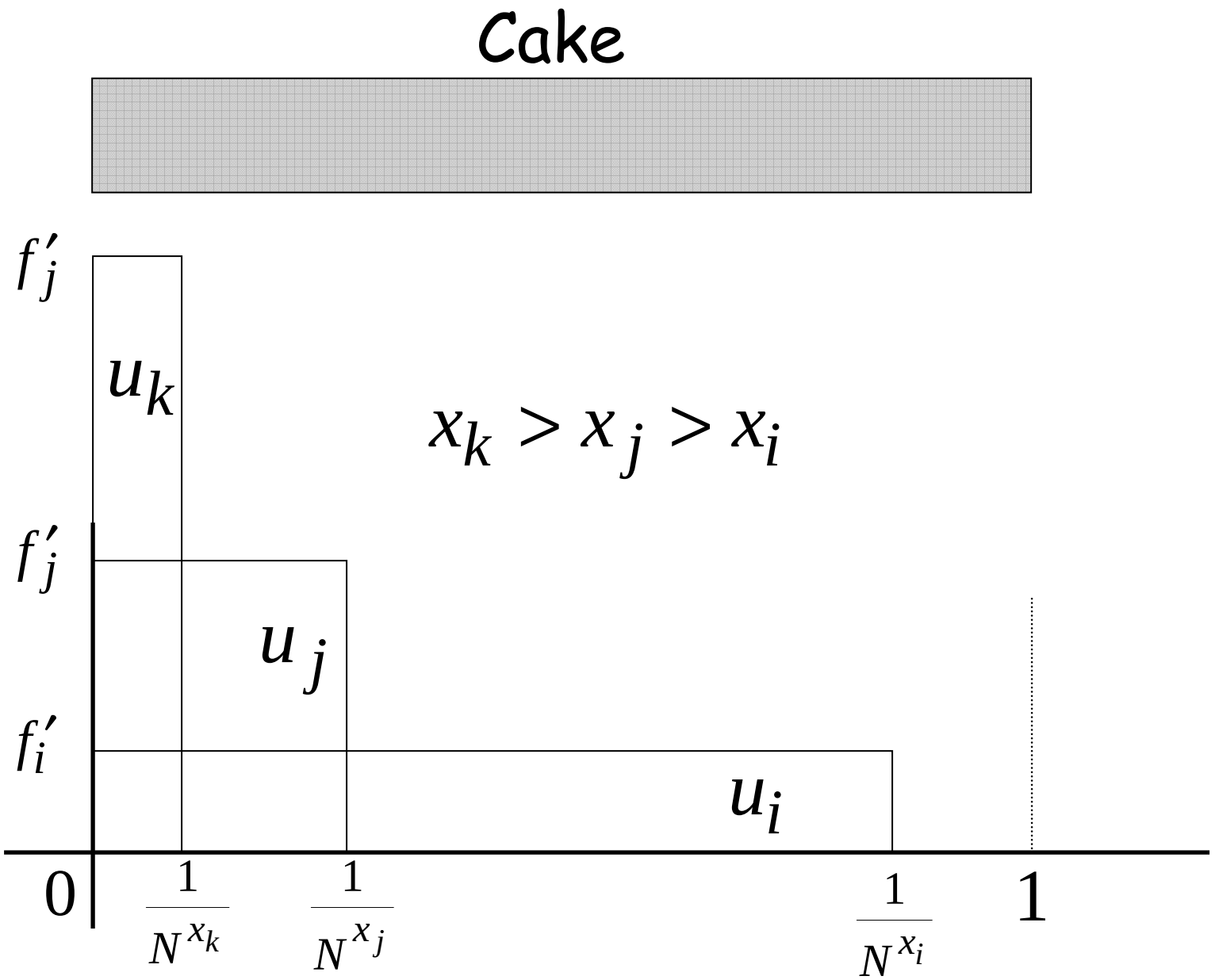
N users: $\quad u_1 \quad u_2 \quad \cdots \quad u_N$

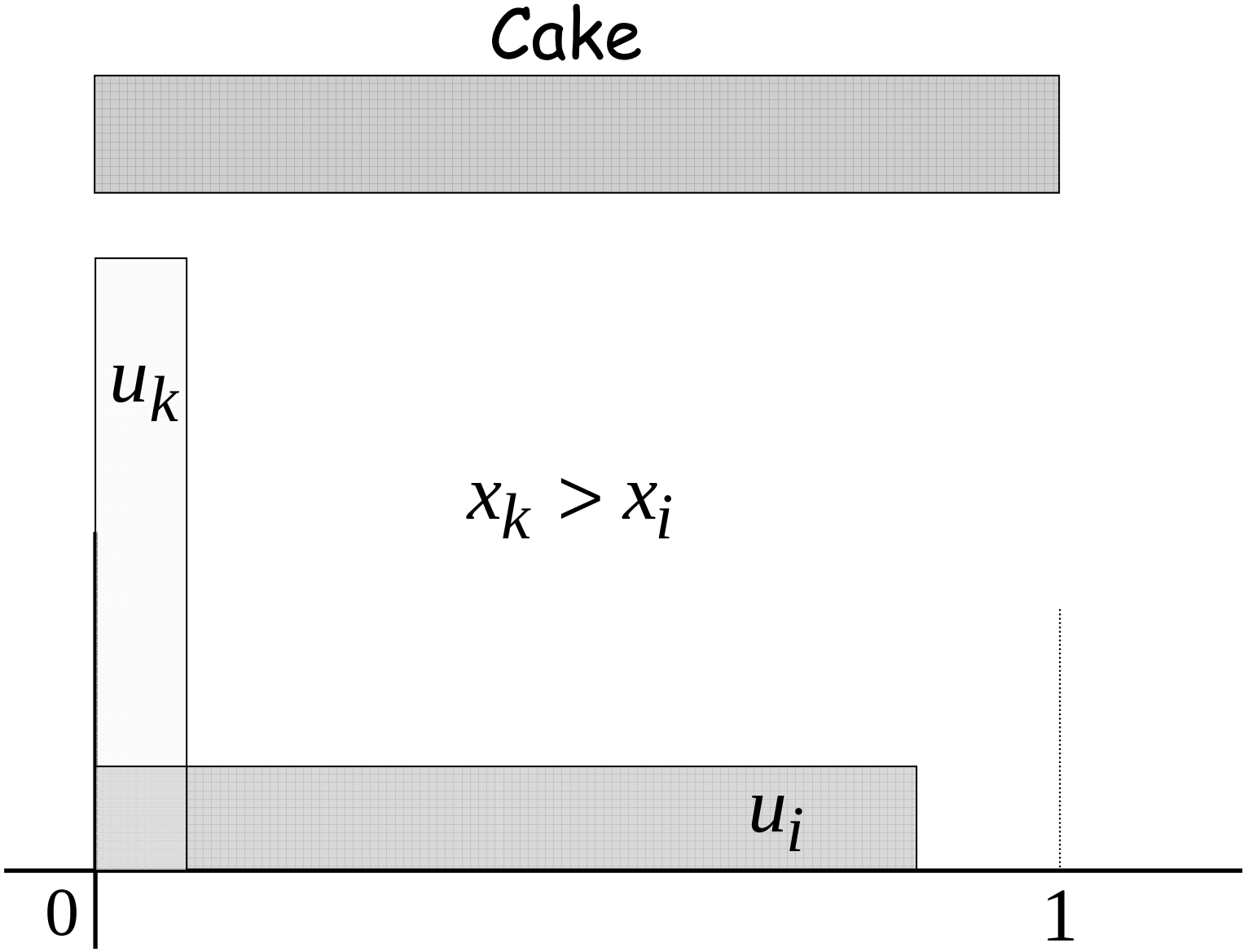
Cake

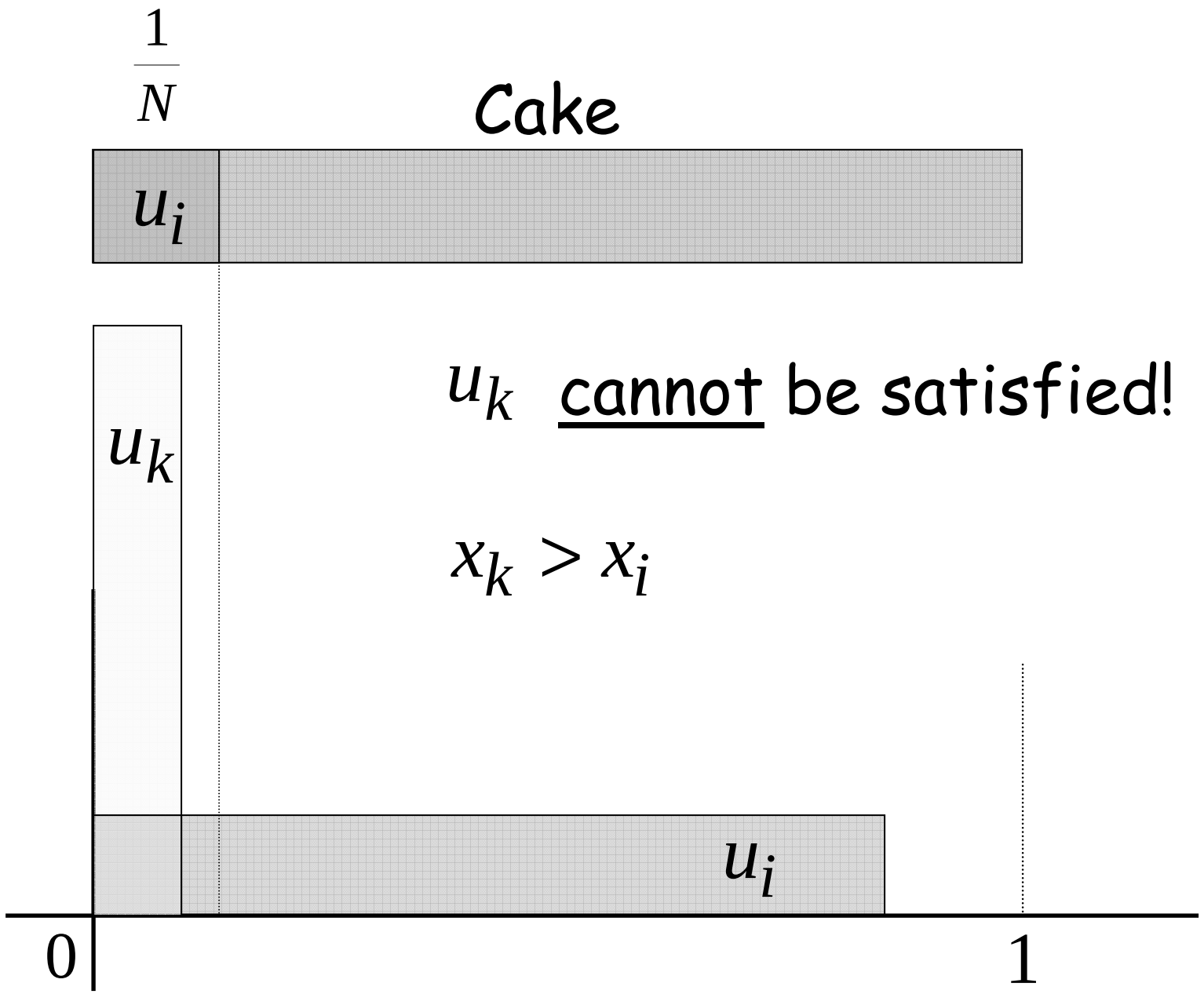


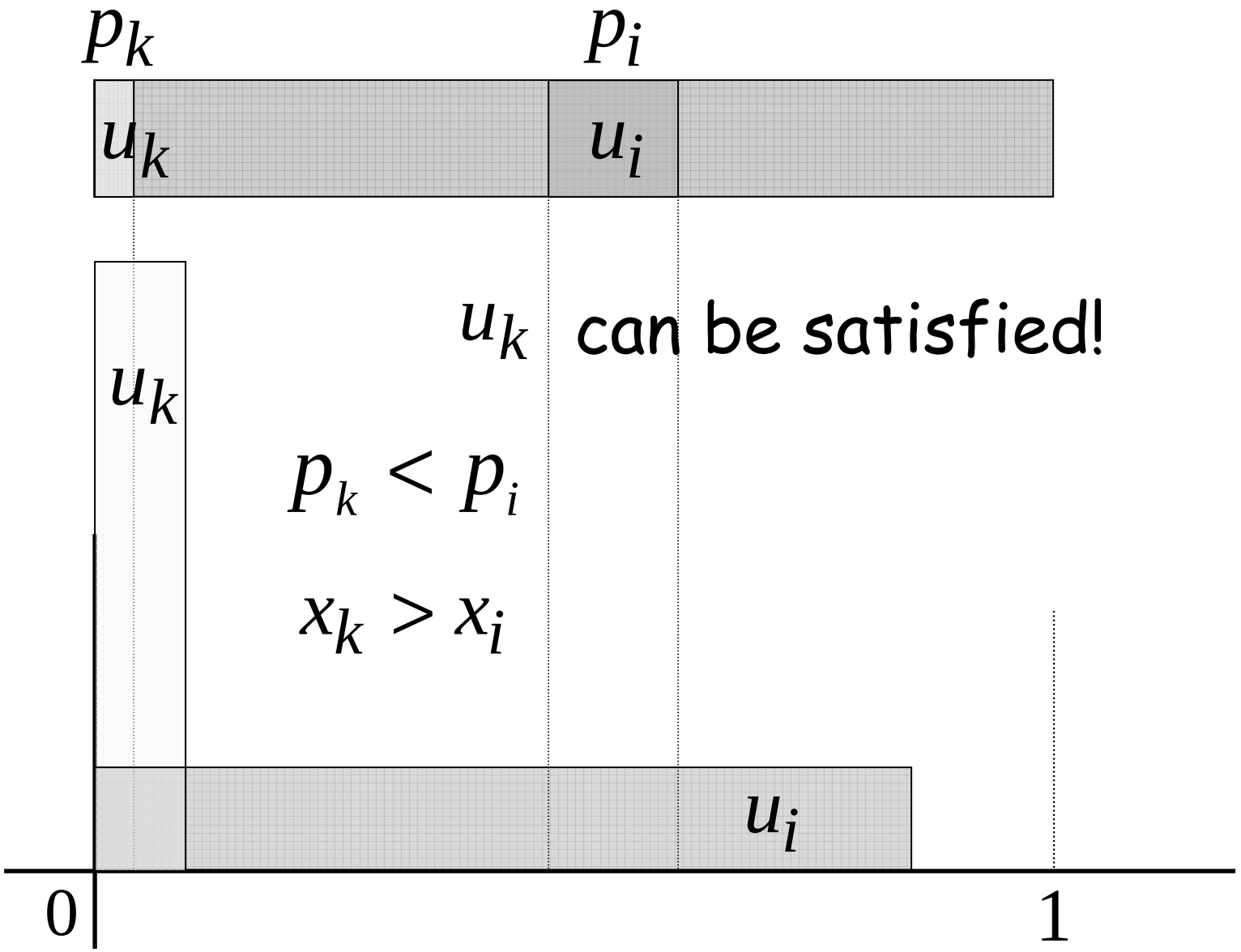
$$f_i(z) = \min(1, N^{x_i} z)$$



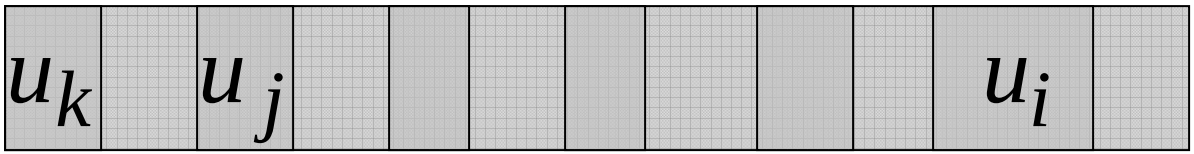

Cake
f_j'
u_k
$x_k > x_j > x_i$
f_j'
u_j
f_i'
u_i
0
$\frac{1}{N^{x_k}}$
$\frac{1}{N^{x_j}}$
$\frac{1}{N^{x_i}}$
1


Cake
u_k
$x_k > x_i$
u_i
0
1


1/N
Cake
u_i
u_k
u_k cannot be satisfied!
$x_k > x_i$
u_i
0
1


p_k
p_i
u_k
u_i
u_k can be satisfied!
u_k
$p_k < p_i$
$x_k > x_i$
u_i
0
1

Cake



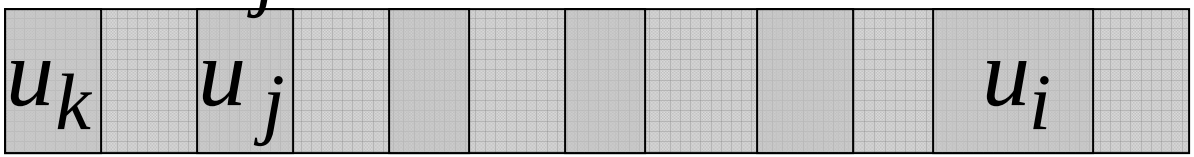
Piece: p_k p_j p_i

Rightmost positive valued pieces

$$p_k < p_j < \cdots < p_i$$

$$x_k > x_j > \cdots > x_i$$

Labels: l_k l_j l_i



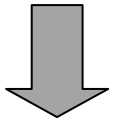
p_k p_j p_i

Sorted labels: $l_k < l_j < \cdots < l_i$

Sorted pieces: $p_k < p_j < \cdots < p_i$

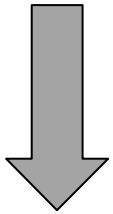
Sorted integers: $x_k > x_j > \cdots > x_i$

Fair cake-cutting



Sorting

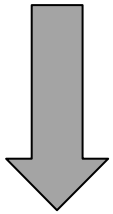
Sorting integers: $\Omega(N \log N)$ comparisons



Cake Cutting: $\Omega(N \log N)$ comparisons

Linearly-labeled &
comparison-bounded algorithms:

Require $O(\#cuts)$ comparisons



$\Omega(N \log N)$ comparisons require

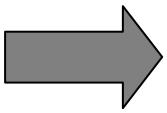
$\boxed{\Omega(N \log N) \text{ cuts}}$

Talk Outline

Cake Cutting Algorithms

Lower Bound for Phased Algorithms

Lower Bound for Labeled Algorithms



Lower Bound for Envy-Free Algorithms

Conclusions

Variations of Fair Cake-Division

Envy-free: Each user feels she gets at least as much as the other users

Strong Envy-free:

Each user feels she gets strictly more Than the other users

Super Envy-free:

A user feels she gets a fair portion,

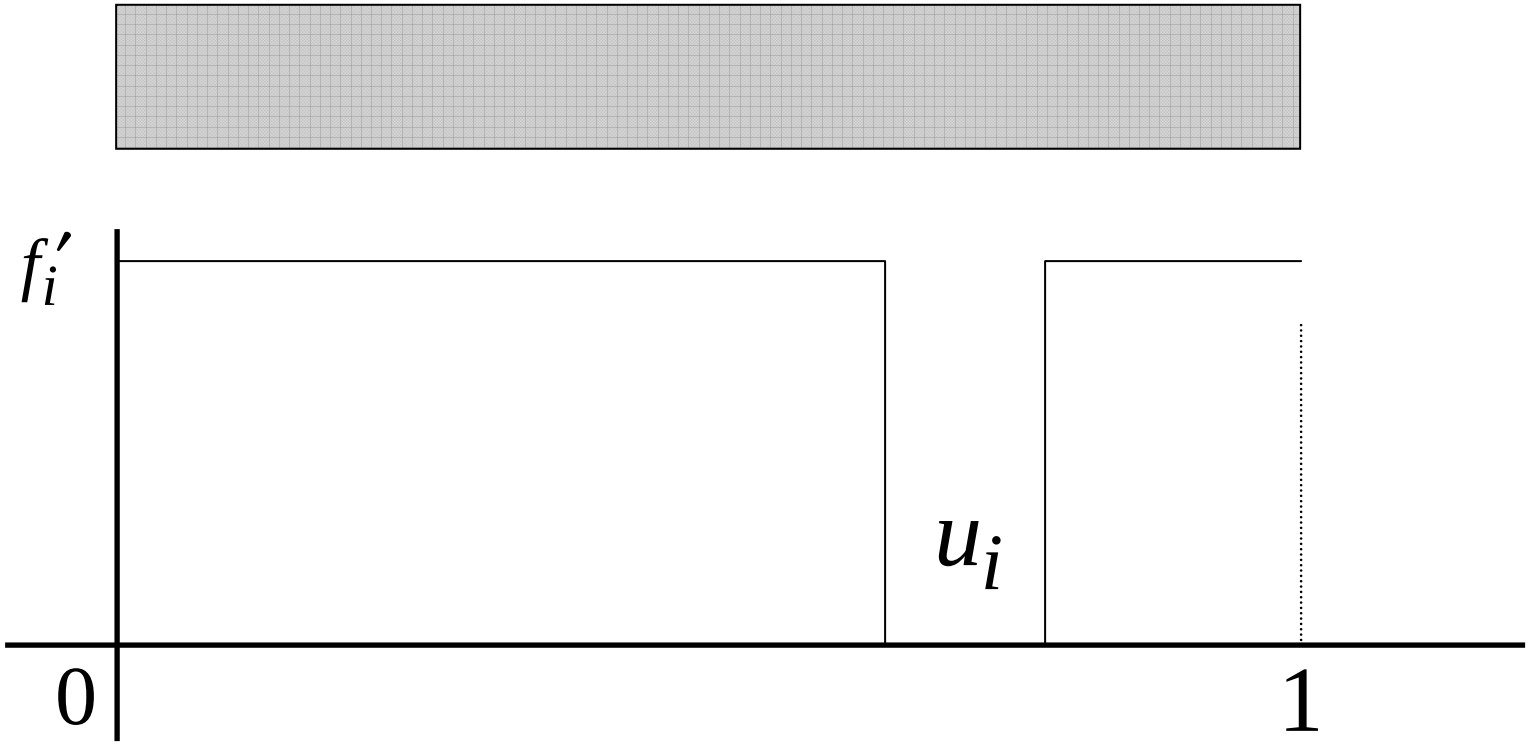
and every other user gets less than fair

Lower Bounds

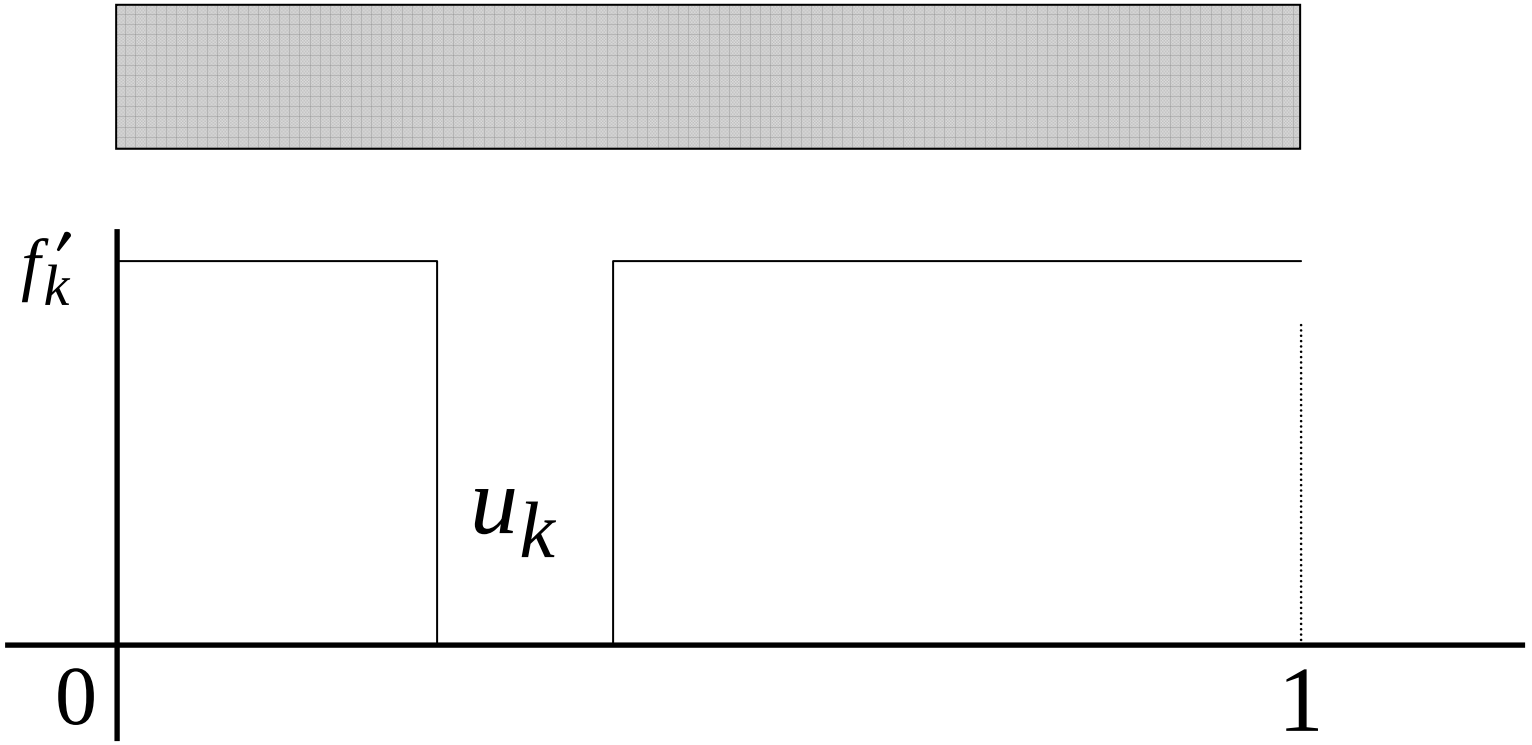
Strong Envy-free: $\Omega(0.086 \cdot N^2)$ cuts

Super Envy-free: $\Omega(0.25 \cdot N^2)$ cuts

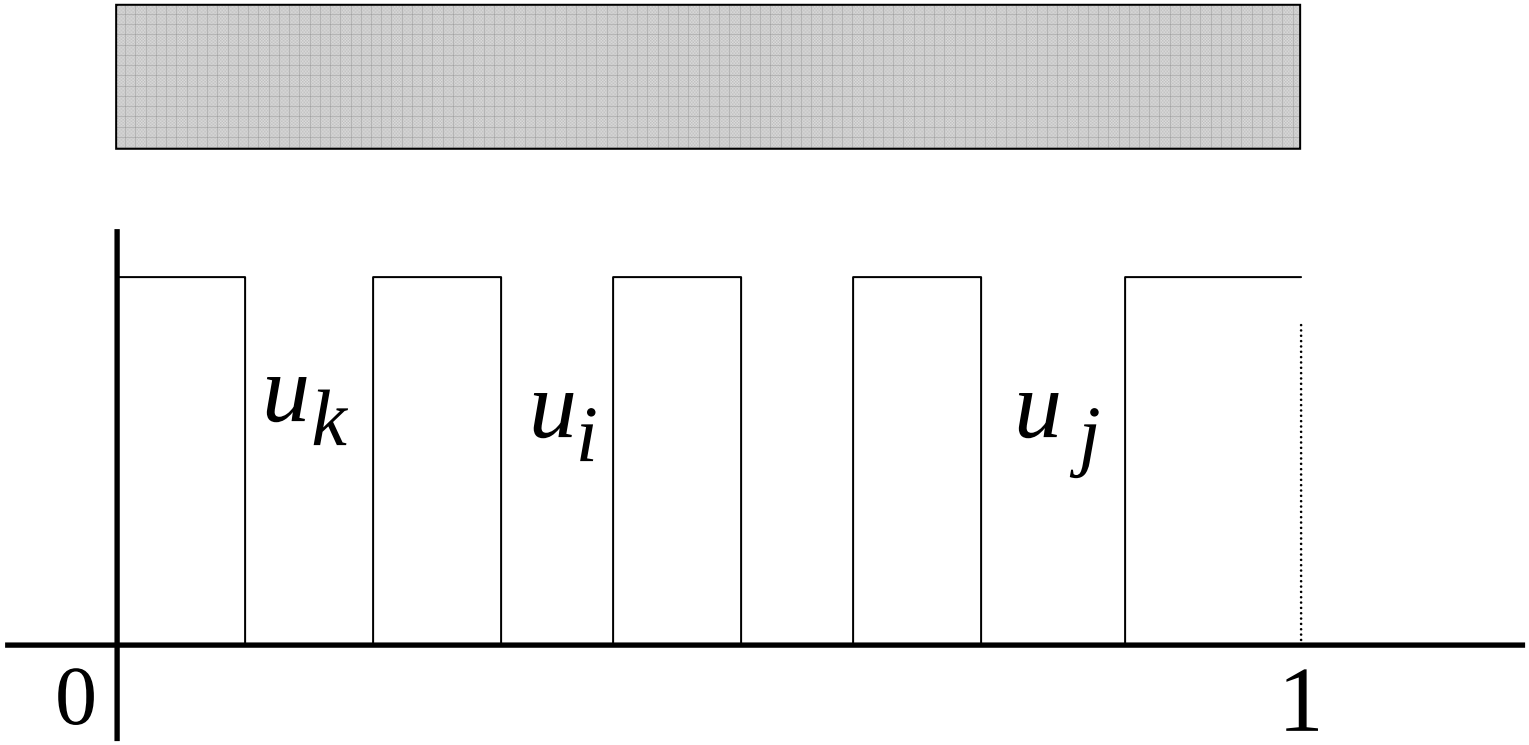
Strong Envy-Free, Lower Bound



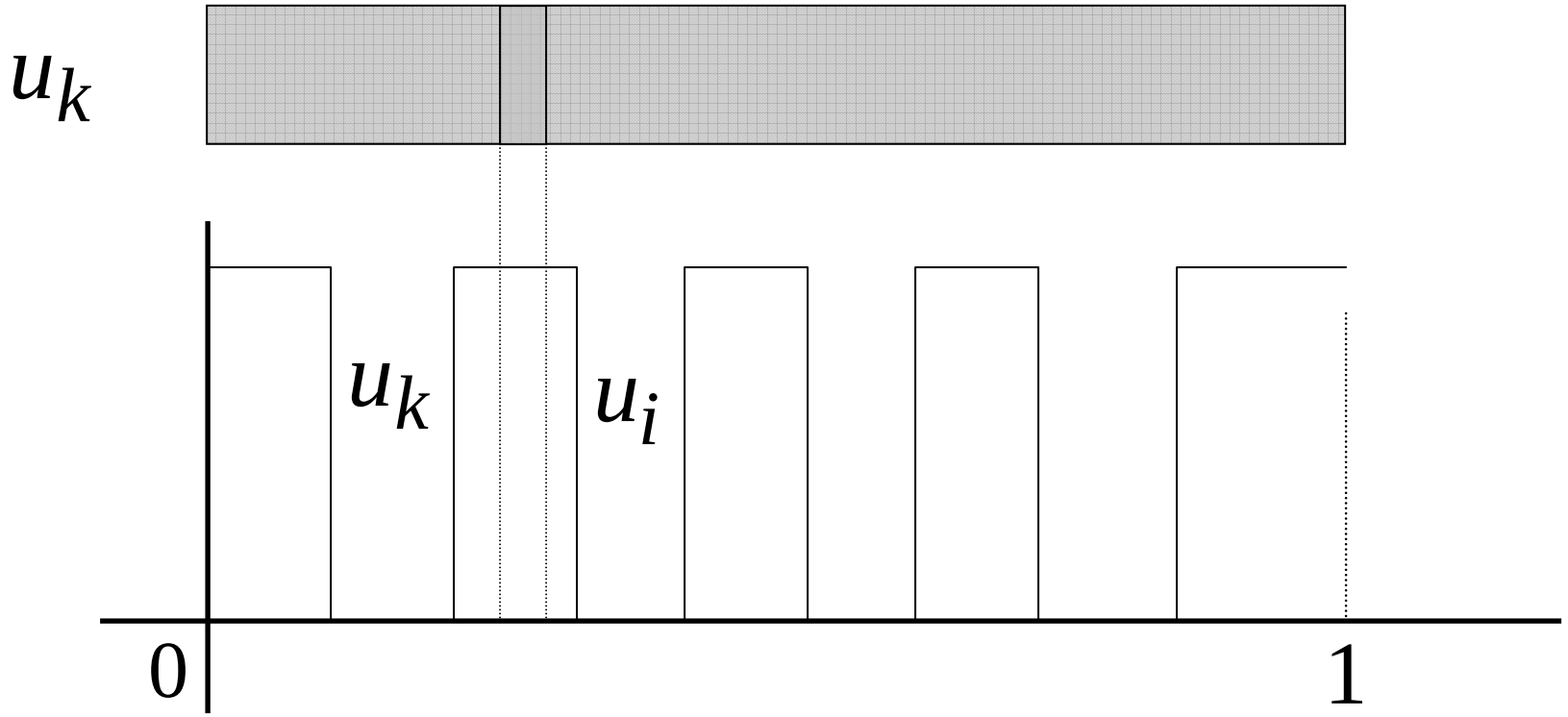
Strong Envy-Free, Lower Bound



Strong Envy-Free, Lower Bound

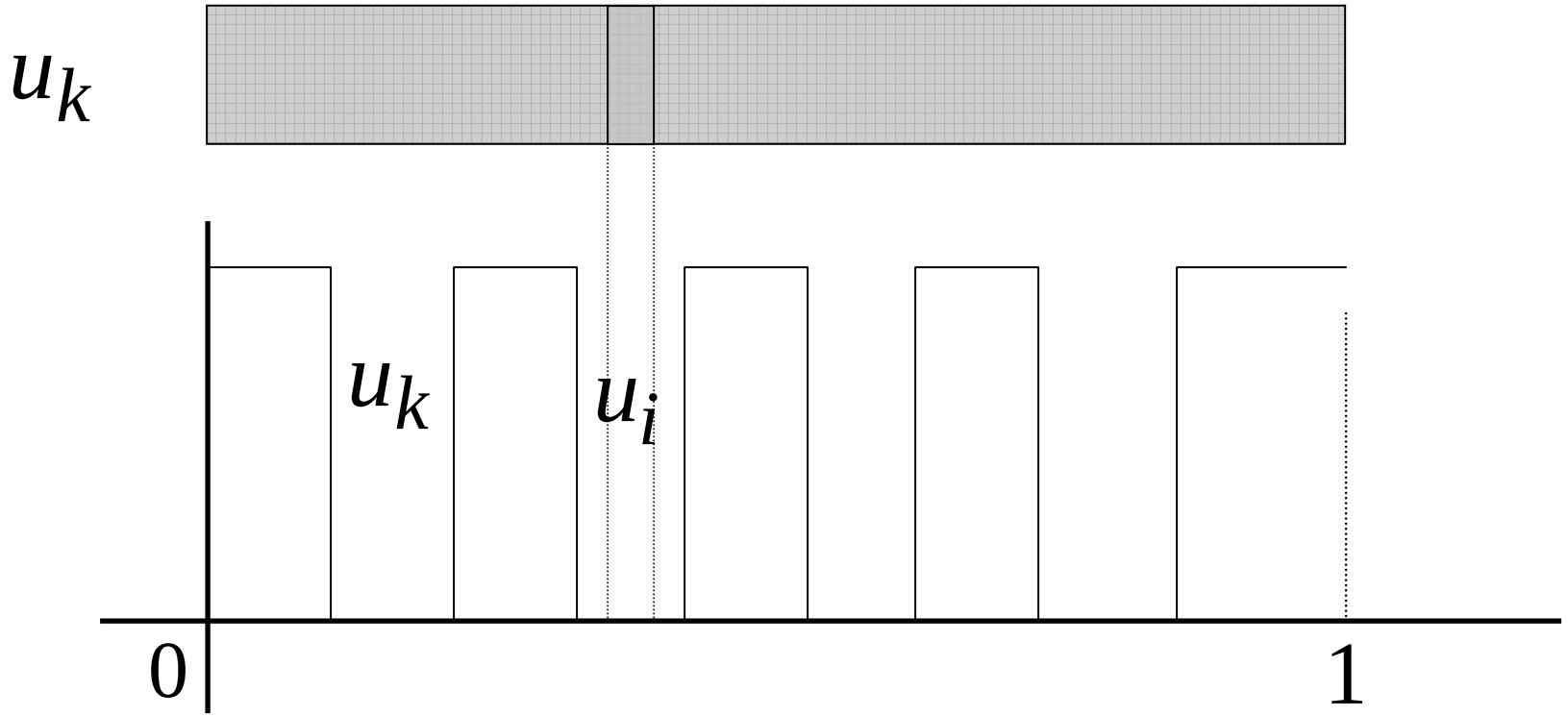


Strong Envy-Free, Lower Bound



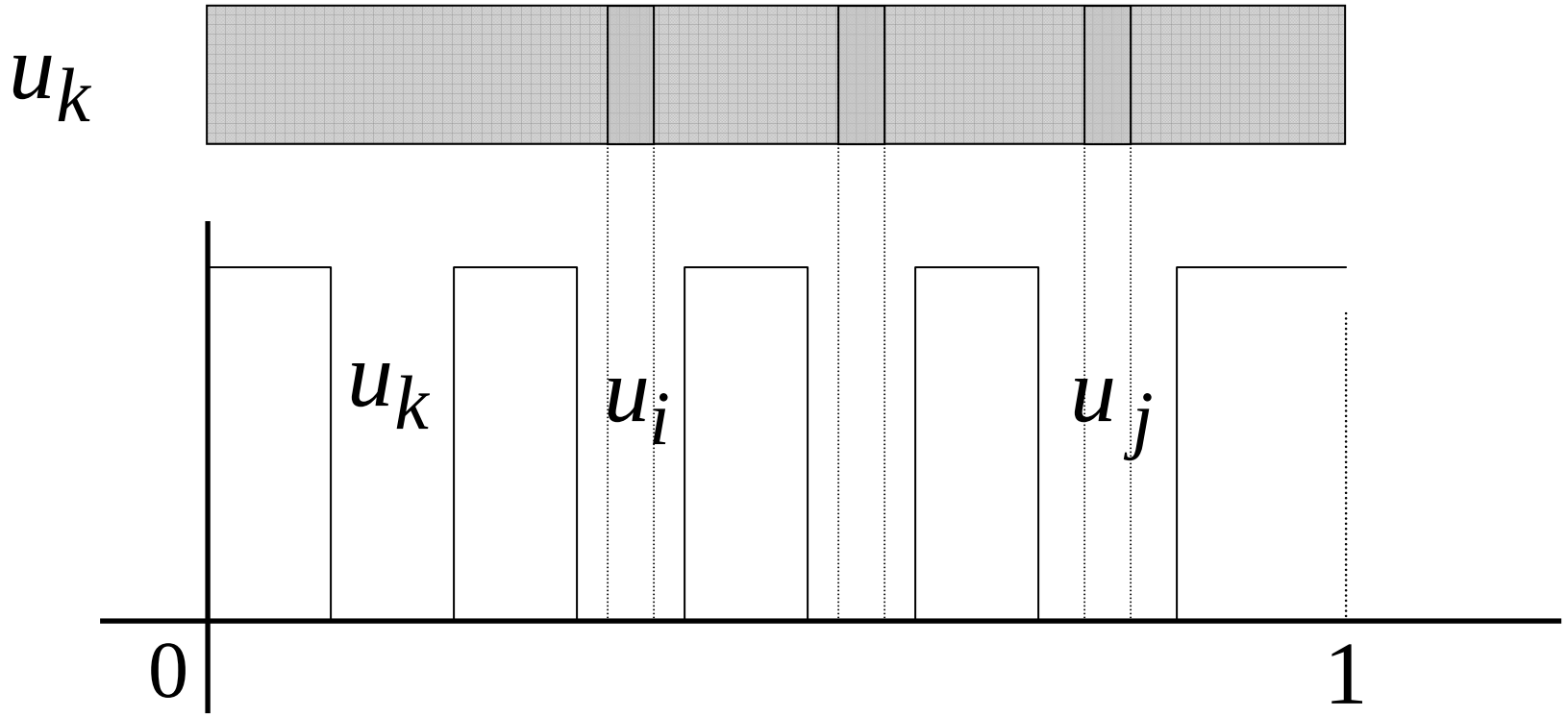
u_k is upset!

Strong Envy-Free, Lower Bound



u_k is happy!

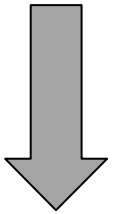
Strong Envy-Free, Lower Bound



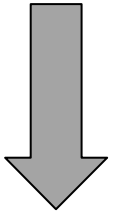
u_k must get a piece from each of the other user's gap

Strong Envy-Free, Lower Bound

A user needs $\Omega(N)$ distinct pieces

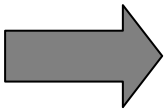


Total number of pieces: $\Omega(N^2)$



Total number of cuts: $\Omega(N^2)$

Talk Outline

Cake Cutting Algorithms
Lower Bound for Phased Algorithms
Lower Bound for Labeled Algorithms
Lower Bound for Envy-Free Algorithms

Conclusions

We presented new lower bounds
for several classes of
fair cake-cutting algorithms

Open problems:

- Prove or disprove that every algorithm is linearly-labeled and comp.-bounded

- An improved lower bound for envy-free algorithms