Balanced Allocation: Patience is not a Virtue

John Augustine[♦], **William K. Moses Jr.**[♦], Amanda Redlich[★],
Eli Upfal[♠]

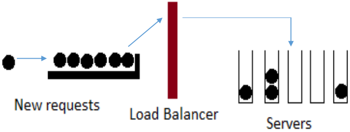
[♦]Indian Institute of Technology Madras, [★]Bowdoin College, [♠]Brown University

SODA 2016

Jan. 11th, 2016

Outline

Load Balancing Problem



Preliminaries

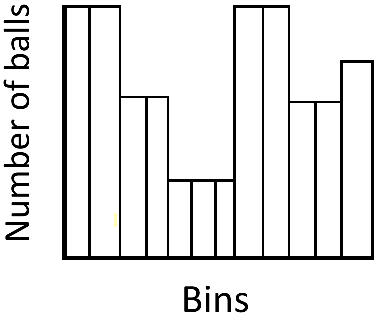
- Balls and Bins model: m balls, n bins, $m \geq n$.
- Sequential ball throwing, one at a time.
- When each ball arrives at the load balancer, loads of bins not known.
- One probe $=$ checking load of one bin.
- Probes made randomly.
- Ball is placed in some bin after suitable number of probes.

Problem Statement

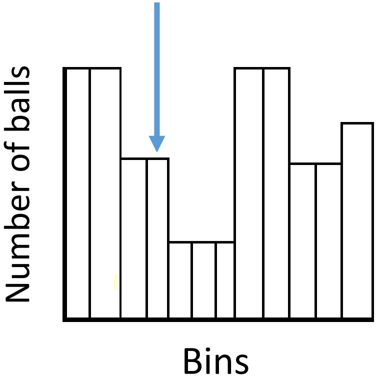
Find an algorithm which minimizes both total number of probes and the maximum load of any bin after all balls are thrown.

Outline

Past Work

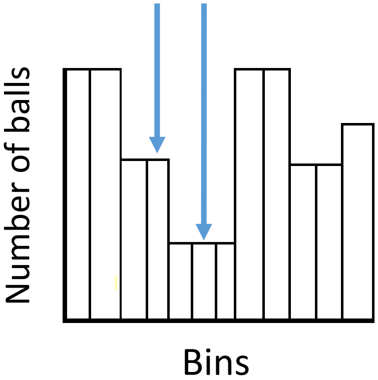


Past Work - Randomly Place each Ball



Max. load of any bin $= \frac{\ln n}{\ln \ln n}(1 + o(1))$ w.h.p. (when $m = n$)

Past Work - Power of Two Choices

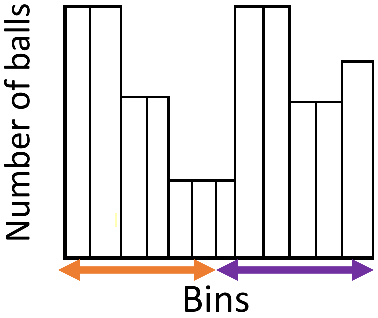


Max. load of any bin $= \frac{m}{n} + \frac{\ln \ln n}{\ln 2} + \Theta(1)$ w.h.p.
[Karp, Luby & Meyer, Algorithmica '96] [Azar, Broder, Karlin & Upfal, SICOMP '99] [Berenbrink, Czumaj, Steger & Vöcking, SICOMP '06]

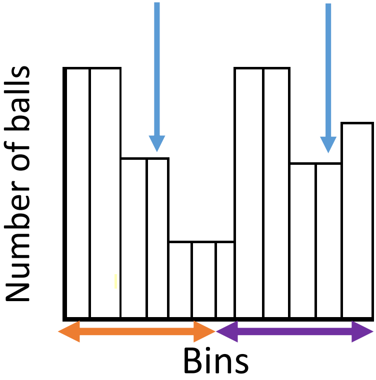
Past Work - Power of d choices aka Greedy[d]

- Power of two choices $\rightarrow$ Power of d choices (Greedy[d])
- Max. load of any bin $= \frac{m}{n} + \frac{\ln \ln n}{\ln d} + \Theta(1)$ w.h.p.
  [Azar, Broder, Karlin & Upfal, SICOMP '99] [Berenbrink, Czumaj, Steger & Vöcking, SICOMP '06]
- Compare with placing ball u.a.r.:
  Max. load of any bin $= \frac{\ln n}{\ln \ln n}(1 + o(1))$ w.h.p. (when $m = n$)

Past Work - Introduce Asymmetry aka Left[2]



Past Work - Introduce Asymmetry aka Left[2]



Max. load of any bin $= \frac{m}{n} + \frac{\ln \ln n}{2 \ln \phi_2} + \Theta(1)$ w.h.p.

[Vöcking, JACM '03] [Berenbrink, Czumaj, Steger & Vöcking, SICOMP '06]

Past Work - Introduce Asymmetry aka Left[d]

- Two choices $\rightarrow$ d choices (Left[d])
- Max. load of any bin $= \frac{m}{n} + \frac{\ln \ln n}{d \ln \phi_d} + \Theta(1)$ w.h.p.
  [Vöcking, JACM '03] [Berenbrink, Czumaj, Steger & Vöcking, SICOMP '06]
- Compare with Greedy[d]:
  Max. load of any bin $= \frac{m}{n} + \frac{\ln \ln n}{\ln d} + \Theta(1)$ w.h.p.

Past Work - Varying the number of probes per ball

- **Idea:** Probe bins until a threshold is found.
- Threshold is a function of maximum number of balls placed.
- *[Czumaj & Stemann, Random Struct. Algorithms '01]*
  - When $m = n$
    Number of probes $= 1.146194m + o(m)$, Max. load of any bin $= 2$ w.h.p.
  - When $m = O(n)$
    Number of probes $= O(m)$, Max. load of any bin $= \lceil \frac{m}{n} \rceil + 1$ w.h.p.
- *[Berenbrink, Khodamoradi, Sauerwald & Stauffer, SPAA '13]*
  - When threshold is a function of ball's placement in input order
    Number of probes $= O(m)$, Max. load of any bin $= \lceil \frac{m}{n} \rceil + 1$ w.h.p.
  - Extending analysis of prior work to $m > n$ case
    Number of probes $= m + O(m^{\frac{3}{4}} \cdot n^{\frac{1}{4}})$, Max. load of any bin $= \lceil \frac{m}{n} \rceil + 1$ w.h.p.

Our Goal

- Get results similar to Left[d].
- Remove - clustering of bins.
- Remove - knowledge of balls' positions in the input order.
- Remove - knowledge of total number of balls to be placed.

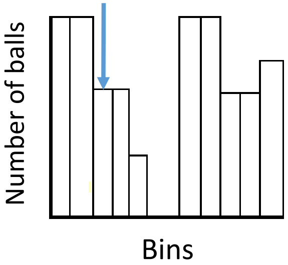
Outline

1. Load Balancing Problem
2. Past Work
3. FirstDiff[d]

FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

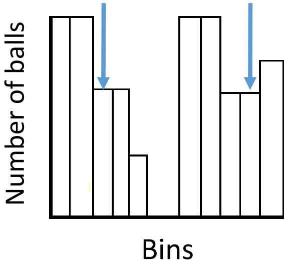
- **First Diff. Condition:** Probe a bin with different load than last seen.



FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

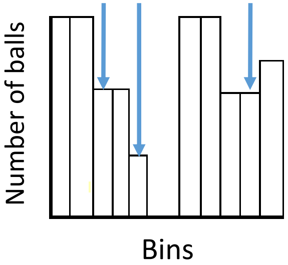
- **First Diff. Condition:** Probe a bin with different load than last seen.



FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

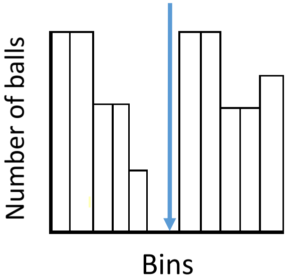
- **First Diff. Condition:** Probe a bin with different load than last seen.



FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

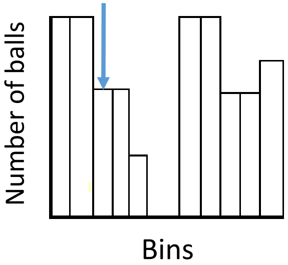
- **Empty Bin Condition:**
  Probe an empty bin.
- **First Diff. Condition:**
  Probe a bin with different load than last seen.



FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

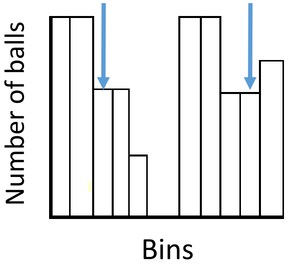
- **Empty Bin Condition:** Probe an empty bin.
- **First Diff. Condition:** Probe a bin with different load than last seen.
- **Flat Bins Condition:** Run out of probes ($2^{\Theta(d)}$ probes allowed per ball, d - average number of probes per ball).



FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

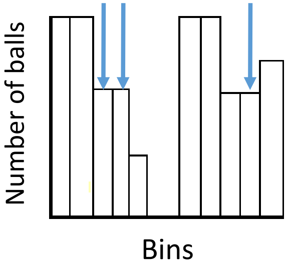
- **Empty Bin Condition:** Probe an empty bin.
- **First Diff. Condition:** Probe a bin with different load than last seen.
- **Flat Bins Condition:** Run out of probes ($2^{\Theta(d)}$ probes allowed per ball, d - average number of probes per ball).



FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

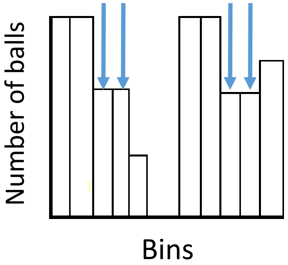
- **Empty Bin Condition:** Probe an empty bin.
- **First Diff. Condition:** Probe a bin with different load than last seen.
- **Flat Bins Condition:** Run out of probes ($2^{\Theta(d)}$ probes allowed per ball, d - average number of probes per ball).



FirstDiff[d] - How it works

Each ball - probe until one of 3 conditions met.

- **Empty Bin Condition:** Probe an empty bin.
- **First Diff. Condition:** Probe a bin with different load than last seen.
- **Flat Bins Condition:** Run out of probes ($2^{\Theta(d)}$ probes allowed per ball, d - average number of probes per ball).



Algorithm

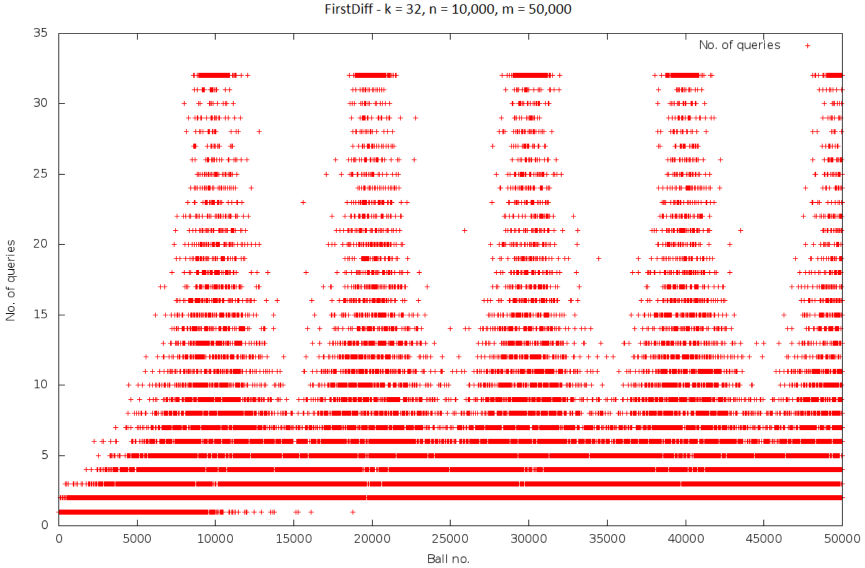
Algorithm 1 FirstDiff[d] (Assume $d \geq 2$. The following algorithm is executed for each ball.)

Repeat $2^{\Theta(d)}$ times
    Probe a new bin chosen uniformly at random
    if probed bin has zero load **then**
        Place ball in probed bin & exit
    if probed bin has load different from those probed before **then**
        Place ball in least loaded bin (breaking ties arbitrarily) & exit
Place ball in last probed bin

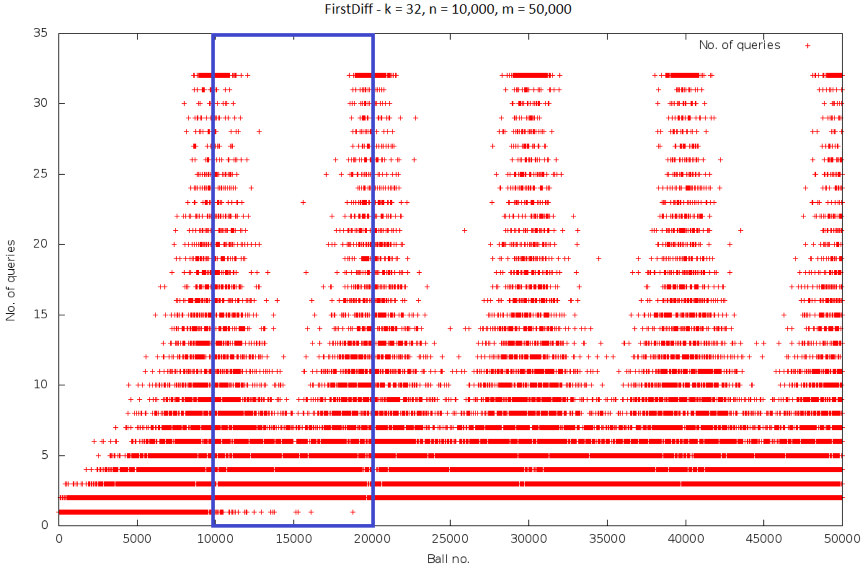
Comparison of Results

- **FirstDiff**[d]
  - Expected number of probes = md.
  - Max. load of any bin ($m = n$) = $\frac{\ln \ln n}{\Theta(d)} + O(1)$ w.h.p.
  - Max. load of any bin ($m \gg n$) = $\frac{m}{n} + \frac{\ln \ln n}{\Theta(d)} + \Theta(\ln \ln \ln n)$ with probability $1 - o(1)$.
- **Comparison:**
  - vs. Greedy[d] - for same expected number of probes, significantly better max. load (Greedy[d] max. load $= \frac{m}{n} + \frac{\ln \ln n}{\ln d} + \Theta(1)$ w.h.p.).
  - vs. Left[d] - for same expected number of probes, similar max. load (Left[d] max. load $= \frac{m}{n} + \frac{\ln \ln n}{d \ln \phi_d} + \Theta(1)$ w.h.p.). But no overhead.
  - Experimentally, when $m = n$, FirstDiff[d] performed better than both Greedy[d] & Left[d].

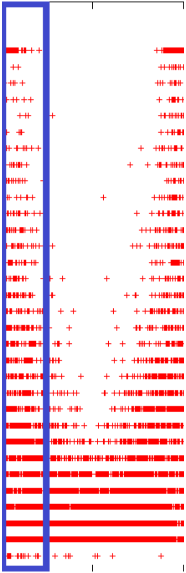
FirstDiff[d] - Number of Probes



FirstDiff[d] - Number of Probes

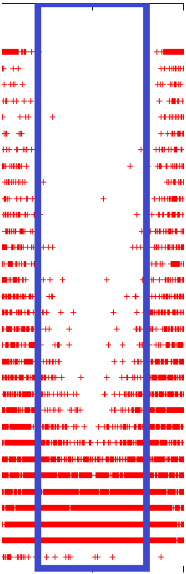


FirstDiff[d] - Number of Probes



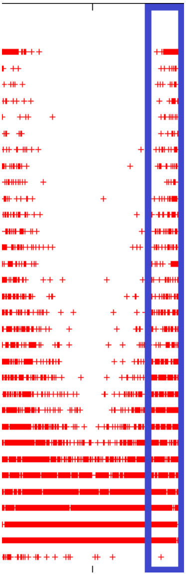
- Let max. number of probes per ball $= k$, i.e. $k = 2^{\Theta(d)}$.
- Expected number of probes per ball $= k$.
- First $\frac{n}{k}$ balls.

FirstDiff[d] - Number of Probes



- Expected number of probes per ball $= \left(\frac{x}{n}\frac{n}{n-x} + \frac{n-x}{n}\frac{n}{x}\right)$.
- Middle $n - 2 * \frac{n}{k}$ balls.

FirstDiff[d] - Number of Probes



- Expected number of probes per ball $= k$.
- Last $\frac{n}{k}$ balls.

FirstDiff[d] - Results

- Number of probes $= O(n \log k) = nd$.
- Proving max. load - layered induction proof.

Extending Results to $m \gg n$ Case

- Max. load
  - Need to handle base case of layered induction when $m \gg n$.
  - Try to avoid any computational component for proof.
  - Start with gap from *[Peres, Talwar & Wieder, SODA '10]*.
  - Use gap reduction lemma from *[Talwar & Wieder, ICALP '14]* to improve gap.
- Number of probes
  - Must capture U-shaped pattern of probes after every n balls placed.
  - Requires us to analyze levels (heights) of balls.

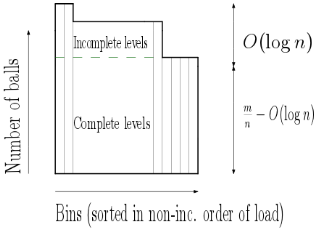
Conclusions

- FirstDiff[d] - Max. load similar to Left[d] without clustering. d probes per ball on average.
- Future - apply FirstDiff[d] to a parallel setting.

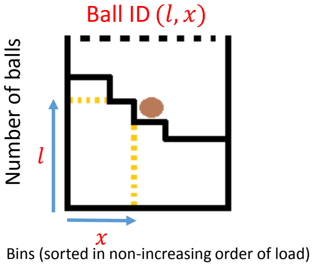
Appendix - FirstDiff[d] - Number of Probes

Result

When $m > n$, expected total number of probes $= md$.



- Let maximum possible probes per ball, $k = 2^{\Theta(d)}$.
- Split balls into complete and incomplete levels.
- We show that number of incomplete levels is $O(\log n)$.
- Each level - at most n balls. Totally $O(n \log n)$ balls.
- Each ball takes at most k probes.
- Expected number of probes to place all balls in incomplete levels $= O(m \log k)$ when $m \geq O(\frac{k}{\log k} n \log n)$.



- Closer look at complete levels.
- Bound expected number of probes for one ball on a given level.
- Sum up expected number of probes for all balls on that level.
- Sum up expected number of probes over all balls of all complete levels.

Appendix - FirstDiff[d] - Number of Probes

- Expected number of probes per level of balls $= O(n \log k)$.
- Number of complete levels $= O(\frac{m}{n} - O(\log n))$
- $\therefore$ Expected number of probes to place all balls in complete levels $= O((\frac{m}{n} - O(\log n))n \log k)$.

Appendix - Tools for Max. Load Proof

- *[Berenbrink, Czumaj, Steger & Vöcking, SICOMP '06]* used computational component in proof of max. load.
- *[Talwar & Wieder, ICALP '14]* provide a tool to simplify proof without a computational component.
- **Tradeoff** - slightly weaker bound (upto $\Theta(\log\log\log n)$).
- **Tool** - Given that there exists a gap between max. load and average load at some time t. **Gap reduction lemma** reduces this gap under some conditions.

Appendix - FirstDiff[d] - Max. Load

Result

When $m > n$,

max. load of any bin $= \frac{m}{n} + \frac{\log \log n}{\Theta(d)} + \Theta(\log \log \log n)$ with probability $1 - o(1)$.

Proof Sketch

- G^t - gap b/w max. loaded bin and average load after tn balls placed.
- Theorem from *[Peres, Talwar & Wieder, SODA '10]* - loose upper bound on G^t for arbitrary t.
- Adapt gap reduction lemma from *[Talwar & Wieder, ICALP '14]*.
- Start at G^t, reduce gap twice to required value.
- Use lemma from *[Talwar & Wieder, ICALP '14]* to s.t. gap holds for all values of t.
- Hence required bound on max. load proved.