Power of Choices: Performance under Error and Compression

Soumya Basu and Vatsal Shah

December 7, 2015

1 Introduction

We consider the balls-into-bins problem where m balls are dropped sequentially in n bins using some randomized scheme. The design of a simple algorithm which can efficiently balance the load among the n bins. This simple model finds its utility in various important applications like hashing, online load balancing, routing, to name a few. The balls generally model independent tasks and items in the system whereas the bins represent the necessary resources.

An important class of randomized allocation schemes is the d-choice, $d > 0$ being an integer. At every step d bins are chosen independently and uniformly at random (i.u.a.r.) from the n bins and dropped into the least loaded of the chosen bins. The scheme with $d = 1$ is known as the "single choice scheme" and has been analyzed extensively in the literature. An interesting result dictates for $m = n$ the heavily loaded bin contains $O(\frac{\log n}{\log \log n})$ (see [5]) balls with high probability. The schemes with $d \geq 2$ for $m = n$, surprisingly achieves a heavily loaded bin of size $O(\log \log n)$ as shown by Azar et al. in their seminal paper [1]. This technique, popularly known as the "Power of Choices", has been explored and extended in various directions including the application in queuing networks [6].

In this review paper we focus on the scaling and the robustness of the balls-into-bins problem motivated from its application in real-world systems. In a practical load balancing scenario the tasks keep on arriving and as a consequence the analysis for $m = n$ fails to capture the evolution of the system. We address this issue in Section 2, where we review techniques which provide guarantees as the number of balls m scales. Further insights from practical systems dictates an algorithm to be robust against errors. The single choice scheme demonstrates robust behavior as the algorithm is independent of the system state. Unfortunately, for $d \geq 2$ the dependence on system state, though small, allows for errors. We investigate this robustness issue in Section 3. The systems in use today consists of large scale resources and it is desired to efficiently store the system state, often using lossy compression techniques. In Section 4 we present recent result which focuses on the space requirements necessary for the implementation of balls-into-bins algorithms. Finally, we conclude in Section 5 following a short discussion in Section 6 about possible future directions in the quest of robust, scalable and simple algorithm for balls-into-bins problem.

2 Balanced Allocation in Balls-into-Bins

In this section we first introduce the model of balls and bins formally for d-choice paradigm. The focus of the remaining part is to analyze the growth in the size of the maximum bin as the number of balls m scales. For example, the analysis of $d = 1$ choice allocation for a broad range of m is considered [5]. We will investigate the maximum load for $d > 1$, i.e. "Power of choices" paradigm, for $m >> n$. A brilliant result due to Berenbrink et al [3], shows that even with $m >> n$ the maximum load is $\frac{m}{n} + \frac{\log \log n}{\log d} + \Theta(1)$ with probability (w.p.) $1 - l\frac{1}{1/poly(n)}$. In this paper we instead provide proof sketch to a relaxed version due to Talwar et al. [7]. This states that the maximum load of a bin after dropping m balls is $\frac{m}{n} + \frac{\log \log n}{\log d} + \Theta(\log^{(3)} n)$ w.p. $1 - \frac{1}{poly(\log n)}$.

2.1 System Model

We consider the weighted balls-into-bins problem as the generalization of the original problem where each ball has a weight w drawn from a distribution $\mathcal{D}$ i.u.a.r. and the load of a bin equals the sum of weights of balls inside the bin at any instance. Clearly, the unweighted case corresponds to the distribution $\mathbb{1}_{\{w=1\}}$.

We consider the system is slotted and at any time slot one ball enters into one bin, following a fixed balls-into-bins algorithm, $\mathcal{A}$. We call an algorithm $\mathcal{A}$ randomized if the algorithm uses random bins in its execution. A generalized framework to describe a large class of randomized algorithm $\mathcal{A}$, uses the *strategy vector* $\mathbf{p} = (p_1, \ldots, p_n)$ where p_i denotes the probability of picking the i^{th} loaded bin in the system (see, [7]). As for example we can describe the single choice algorithm $\mathcal{A}_1$ as $\mathbf{p} = \frac{1}{n}\mathbf{1}$, where as the d-choice algorithm $\mathcal{A}_d$ corresponds to the probability vector,

$$p_i = \left(\frac{i}{n}\right)^d - \left(\frac{i-1}{n}\right)^d, \forall\, i \in [n] \tag{1}$$

The system can be represented as a normalized vector $\mathbf{x}(t) = \{x_1(t), \ldots, x_n(t)\}$, where $x_i(t)$ represents the deviation from the mean load in the bin i at time slot t. Further it has been assumed that $\mathbf{x}$ has been sorted in increasing order $x_1 \geq \cdots \geq x_n$. Denote the random process of the load deviation in bin under allocation under $\mathcal{A}$ as $\mathbf{X}(t)$. For a *strategy vector* $\mathbf{p}$ the process $\mathbf{X}(t)$ is given by the following discrete time Markov process (DTMC) [7],

- Given the current state $\mathbf{x}(t)$ for the ball at $(t+1)$-th step
  - Sample the new position, $j \in_{\mathbf{p}} [n]$
  - Sample the weight, W from distribution $\mathcal{D}$
- Define y_i as follows, for all $i \in [n]$:

$$y_i = \begin{cases} x_i(t) + W - \frac{W}{n} \text{ if } i = j \\ x_i(t) - \frac{W}{n} \text{ if } i \neq j \end{cases}$$

- Sort $\mathbf{y} = (y_1, \ldots, y_n)$ to obtain new state $\mathbf{x}(t+1)$

For the rest of the section we are interested in establishing high probability upper bounds for the highest deviation from the mean, $Gap(t) = \max_i x_i(t)$. It is obvious that any such bound on $Gap(t)$, say $G_n(t)$, implies the heaviest loaded bin contains $\frac{t}{n} + \Theta(G_n(t))$ w.h.p.

2.2 Power of Choices

In this section we address heavily loaded balls-into-bins problem with Power of Choices. The concentration around the mean load for unweighted balls was first shown by Berenkbrink et al. [3]. Recently, Talwar et al. extended the result for the weighted balls [7]and provided a simpler proof for the unweighted version with slightly relaxed concentration [8].

Theorem 2.1. *For any m, the gap between maximum and average of Power of d choices for unweighted balls, follows*

- *$Pr[Gap(m) > \frac{\log \log n}{\log d} + \gamma(c)] \leq n^{-c}$, for any $c > 0$ [3],*

- *$Pr[Gap(m) > \frac{\log\log n}{\log d} + c\log^{(3)} n] \leq n^{-c}$, for some absolute constant c [8].*

Proof. The proof of high probability bounds on the $Gap(m)$ for $m >> n$ is built upon two key ideas, namely,

- **Weak Gap Theorem** which states that w.h.p. $Gap(t)$ is bounded reasonably for n and t.
- **Short Memory Theorem** which proves given an initial gap of Δ, additional poly$(n)\Delta$ steps of the "Power of Choices" allocation wipes out the memory of the system.

In the rest of the section we develop the key blocks in the proof differentiating the weighted case from the unweighted case, whenever necessary. Further for this purpose, let us denote the k-th moment of weight distribution $\mathcal{D}$ as M_k and it is finite for $k = 2$.

Before presenting the Weak Gap Theorem it is of some value to differentiate the approach taken in [3] to that of [8]. In the work of Berenbrink et al. the Weak Gap Lemma statement is somewhat trivial. This leads to the problem of tight analysis of the Short Memory Theorem where the authors needed to account for even a constant number of ball mismatch in n drops of balls. On the other hand using potential function approach Talwar et. al. were able to reduce the efforts needed in the Short Memory Theorem part. We follow the approach of Talwar et al in this paper. Specifically, they draw results from [4] to construct a potential function of the form $\sum_i exp(a|x_i|)$ and using Lyapnov drift technique show that for $d > 1$,(a and b are constants)

$$\mathbb{E}\left[\sum_i exp(a|x_i|)\right] \leq bn. \tag{2}$$

Theorem 2.2. *Weak Gap Theorem [8] For all t and every $c \geq 0$, it holds that $Pr[Gap(t) \geq \frac{c\log n}{a}] \leq \frac{bn}{n^c}$. This implies for every c there is a γ such that $Pr[Gap(t) \geq \gamma \log n] \leq n^{-c}$.*

Proof. Follows from Markov inequality for $\sum_i exp(a|x_i|)$ and inequality 2. □

We next present the Short Memory theorem of [8] which implies that from a given gap Δ it takes $n\Delta$ more steps to attain a gap of $O(\log L)$ on an average. It can be easily seen that the Theorem 2.2 provides a nice starting point of $L = O(\log n)$ which allows us to move to a gap of $O(\log\log n)$ in $O(n\log n)$ steps. Finally, to attain a high probability bound as given in Theorem 2.1 ([8] version), another $O(n\log\log n$ suffice. Next we present the Short Memory Theorem for $d = 2$ which easily generalizes to $d > 2$.

Theorem 2.3. *Short Memory Theorem [8] There is a universal constant γ such that the following holds: for any t, ℓ, L such that $1 \leq \ell \leq L \leq n^{\frac{1}{4}}$ and $Pr[Gap(t) \geq L] \leq \frac{1}{2}$,*

$$Pr[Gap(t+L) \geq \log\log n + \ell + \gamma] \leq Pr[Gap(t) \geq L] + \frac{16bL^3}{exp(a\ell)} + \frac{1}{n^2},$$

where a and b are given in inequality 2.

Proof. The proof uses the classic Layered Induction approach from Azar et al. [1]. The first term in the bound, $Pr[Gap(t) \geq L]$, represents bad event in the starting case. Conditioned on the good events $Gap(t) < L$ it argues that the probability of bins which contain large number of bins drops doubly exponentially in the number of steps t. But this does not suffice to extend the bounds to $t > n$. Instead using the fact even the lightly loaded bins are concentrated around the mean one can argue as the system enter the regime of $t > n$ the lightly loaded bins gets picked more. □

□

Finally to conclude this section we present the generalization to the weighted balls case. As the authors of the paper [8] remark that all the components of the proof extends to weighted case except for the necessary condition $Pr[Gap(t) \geq c' \log n]$ to be small enough. This holds for unweighted case through comparison with single choice schemes but breaks down for weighted case for certain regime of m. Specifically, for $O(n)$ and $\Omega(n \log n)$ the bounds in Theorem 2.1 is valid for the weighted case as well. They provide matching lower bonds on the gap for the mentioned regime. Further the authors show that for certain distribution outside the mentioned regime we have $\Omega(\log n)$ gap.

3 Error in Choices: $(1 + \beta)$ choice

Until now, we have discussed the balls and bins problem under the standard assumptions that the information about the load of any bin is instantaneously and accurately available to every incoming ball. The breakdown of these assumptions does not affect the single choice problem but it does cause issues in the $d-choice$ problem. Incorrect reporting of the load can be attributed to a host of factors:

- Errors may be introduced while communicating the load
- Delays in the system means that the load reported by the system at the current time slot can be inaccurate
- For some cases, it might not be practical to store the load of all the bins when n is very large and so communicating the load of each and every bin may not be always possible

Let us define 'gap' as the difference between the maximum load and the average load.

There are m balls and n bins and each ball falls in a random bin with probability $1 - \beta$, and the lesser loaded of two (d for d-choice) random bins with probability β. Thus, when $\beta = 0$, this problem reduces to the single-choice problem and when $\beta = 1$ it reduces to two choice problem. For the standard case, we know that when its a single choice problem, the gap is $O(\sqrt{\frac{m \log n}{n}})$ with high probability while for the $d - choice$ problem the gap is atmost $O(\frac{\log \log n}{\log d})$ with high probability. The work by [4] deals with finding the bounds on the maximum load for the $(1+\beta)$ choice problem which is an intermediate case between the above two problems.

The *strategy vector* for the $(1 + \beta)$ choice problem is given as follows:

$$p_i = \frac{\beta(2i - 1)}{n^2} + \frac{1 - \beta}{n} \tag{3}$$

As has been mentioned in section 2, the monotonicity of p_i is necessary for showing the upper and lower bounds for the maximum loaded bin. To find the upper bound, this paper draws parallels with the biased random walk in d dimensions (for d choice problem) and finds the expected maximum drift (load) with high probability. We know that an unbiased random walk in two dimensions starting at 0 is at an expected distance of $\sqrt{m}$ after m steps. But if the bias is towards 0, then the expected distance is a constant independent of m. This will be our motivation to obtain an upper bound for the $(1 + \beta)$ choice problem.

For our case, we know that the difference between two loaded bins is an unbiased random walk with probability $1 - \beta$ and a biased random walk with probability β. In order to show results using mean drift criteria, we need to define a suitable potential function Γ which satisfies the following conditions:

i) If Γ is small then the allocation is balanced

ii) Expectation of Γ is independent of m

In this section, we will give a sketch of the proof for the high probability upper bounds of the maximum load in the $(1+\beta)$ process with weighted balls for a large class of distributions $\mathcal{D}$. The system model described in the Section 2.1 is adopted in the following analysis.

The authors make the following two assumptions regarding the elements of the probability distribution vector $\mathbf{p}$:

a) $p_i \leq p_{i+1}\ \forall i \in [n-1]$: Ensures that the performance of the $(1+\beta)$ choice is not worse than the single choice method

b) $p_{\frac{n}{3}} \leq \frac{1-4\epsilon}{n}$ and $p_{\frac{2n}{3}} \geq \frac{1+4\epsilon}{n}$: Guarantees that the allocation rule prefers the least loaded third of bin over the most loaded third

For $\epsilon = \frac{\beta}{12}$, both these assumptions are satisfied for the $(1+\beta)$ choice process . The authors in [4], then define the moment generating function $M(\lambda) = E[\exp(\lambda W)]$ and there exists a $z > 1$ such that for every $|z| < \frac{\lambda}{2}$, there is a $S \geq 1$ such that $M"(z) < 2S$.
The potential function is then defined as follows:

$$\Gamma(t) = \Gamma(x(t)) := \Phi(t) + \Psi(t)$$
$$\text{where } \Phi(t) = \Phi(x(t)) := \sum_{i=1}^{n} \exp(\alpha x_i)$$
$$\text{and } \Psi(t) = \Psi(x(t)) := \sum_{i=1}^{n} \exp(-\alpha x_i)$$

where $\alpha = \min \frac{\epsilon}{6S}, \frac{\lambda}{2}$

The proof mainly involves using the Lyapunov drift technique for the above potential function $\Gamma(t)$. A brief summary of the proof including the following three lemmas used for proving the high probability upper bounds for the $1+\beta$ choice problem is now described in the remaining part of this section:

Lemma 3.1. $\mathbb{E}\left[\Phi(t+1) - \Phi(t)|x(t)\right] \leq \sum_{i=1}^{n}\left(p_i(\alpha + S_{\alpha}^{2}) - (\frac{\alpha}{n} - S\frac{\alpha^2}{n^2})\right)\exp\left(\alpha x_i\right) \leq \frac{2\alpha}{n}\Phi(t)$
$\mathbb{E}\left[\Psi(t+1) - \Psi(t)|x(t)\right] \leq\leq \sum_{i=1}^{n}\left(p_i(-\alpha + S_{\alpha}^{2}) + (\frac{\alpha}{n} + S\frac{\alpha^2}{n^2})\right)\exp -(\alpha x_i) \leq \frac{2\alpha}{n}\Psi(t)$

Lemma 3.2. *If* $x_{\frac{3n}{4}}(t) \leq 0$, *then* $\mathbb{E}\left[\Phi(t+1)|x(t)\right] \leq \left(1 - \frac{\alpha}{n}\right)\Phi(t) + 1$
If $x_{\frac{n}{4}}(t) \geq 0$, *then* $\mathbb{E}\left[\Psi(t+1)|x(t)\right] \leq \left(1 - \frac{\alpha}{n}\right)\Psi(t) + 1$

Lemma 3.3. *Suppose that* $x_{\frac{3n}{4}} > 0$ *and* $\mathbb{E}[\Delta\Phi|x(t)] \geq \frac{-\alpha\epsilon}{4n}\Phi$. *Then either* $\Phi < \frac{\epsilon}{4}\Psi$ *or* $\Gamma < cn$
Suppose that $x_{\frac{n}{4}} < 0$ *and* $\mathbb{E}[\Delta\Psi|x(t)] \geq \frac{-\alpha\epsilon}{4n}\Psi$. *Then either* $\Psi < \frac{\epsilon}{4}\Phi$ *or* $\Gamma < cn$

3.1 is used to prove the upper bounds for the expected 1-slot drift for $\Phi(t)$ and $\Psi(t)$. The proofs mainly involve using moment generating functions to bound the drifts and then bounding these moment generating functions based on our assusmptions.

3.2 deal with showing that either $\Phi(t)$ or $\Psi(t)$ decrease in expectation conditioned on $x(t)$ when the configurations are reasonably balanced. The proof for these lemmas involves formulating an optimization problem to find the maximum value of the first term ($p_i(\alpha + S_{\alpha}^{2})\exp\left(\alpha x_i\right)$ and $p_i(-\alpha + S_{\alpha}^{2})\exp\left(-\alpha x_i\right)$ respectively) in Lemma 2.1 with the constraints of reasonbaly balanced configurations and monotonicity of $\exp\left(\alpha x_i\right)$ and then using the maximum value of the optimization problem to obtain the required decrease in expectation.

3.3 is used to bound $\Gamma(t) \leq an$ for some $a > 0$. This is proved by combining the previous two lemmas along with the assumptions of reasonably balanced configurations of the bins. Combining the above three lemmas, we get our main theorem which shows the supermartingale type of property for $\Gamma(t)$,

Theorem 3.4. $\mathbb{E}[\Gamma(t+1)|x(t)] \leq \left(1 - \frac{\alpha\epsilon}{4n}\right)\Gamma(t) + 1$

Using the above theorem, it is easy to show that $\mathbb{E}[\Gamma(t)] = O(n)$ and since we have $\exp(\alpha Gap(t)) \leq \Gamma(t)$, we have the following bound for the gap between the maximum load and average load with high probability:

$$\mathbb{E}[Gap(t)] \leq \frac{1}{\alpha}\mathbb{E}[\Gamma(t)] \leq O\left(\frac{\log \frac{n}{\beta}}{\beta}\right)$$

Using results from [5], the authors similarly show a lower bound for the expected gap, $\mathbb{E}[Gap(t)] = \Omega\left((1-\beta)\frac{\log n}{\beta}\right)$. To summarize, this paper transforms the balls and bins problem into a Markov Chain and then uses the Mean Drift criteria to obtain upper bounds for the maximum load in the bin for the $(1+\beta)$ choice problem. This is the one of the first works on the bounds for the $(1+\beta)$ choice problem for the non-trivial weighted case.

4 Compression

This section mainly deals with the issue of compression in balls and bins. For n balls and n bins, the load in the maximum bin is $O\left(\frac{\log n}{\log\log n}\right)$ for the single choice problem. For the two choice problem, the maximum load in any bin is $O(\log\log n)$. Thus, we will need a maximum of $O(\log\log\log n)$ bits to store the load of each bin and for n bins, we will require $O(n\log\log\log n)$ bits. Compression is necessary when:

- Insufficient Memory: When we do not have sufficient memory to store the load of all the bins.
- Communication complexity: When it is not possible to communicate the load of all the bins.

[1] showed that if we divide the number of bins into groups of $\log\log n$ bins, then we only require $O\left(\frac{n}{\log\log n}\log\log\log n\right) = o(n)$ bits. [2] discusses this issues and gives lower bound when we have $n^{1-\delta}$ bits of memory. The following results hold for the $n - balls$, $n - bins$ case and each ball has 2 choices to make. But the information about the load of the bin may not always be available

Theorem 4.1. *We are sequentially given n balls. We have to put each of them into one of two bins chosen uniformly and independently at random among n bins. We have only $M = n^{1-\delta}$ bits of memory ($\delta > 0$ may depend on n), our choice where to put a ball can depend on these memory bits and random bits. Then the maximum load will be atleast $\delta\frac{\log\log n}{\log n}$ with probability $1 - o(1)$.*

The proof of this theorem can be obtained by applying Chernoff bounds on the events that the ball is put in a new bin instead of those already present in the memory.

The following theorem gives matching upper bounds in the communication complexity model with atmost $n^{1-\delta}$ bits of transmission possible.

Theorem 4.2. *There exists an algorithm that gets $M = n^{1-\delta}$ bits of advice before each step and uses no other memory, and ensure that the heaviest bin contains atmost $O\left(\frac{\delta\log n}{\log\log n}\right)$ balls w.h.p.*

For a two-choice problem, the probability p_i that the ball is in bin i never exceeds $\frac{2}{n}$ and so the probability that after n steps the total number of balls exceeds $T = \frac{2\delta\log n}{\log\log n}\left(1 + \frac{2\log\frac{1}{\delta}}{\log(\delta\log n)}\right)$. This quantity can be asymptotically be upper bounded by $o(\frac{1}{n^{\delta}\log n})$. Thus the number of bins with T balls is atmost $\frac{n^{1-\delta}}{2\log n}$ *w.h.p.*. For the upper bound, the authors suggest a new algorithm. The algorithm suggested in [2] basically only communicates the load of all the bins in L, i.e. the set of bins with atleast T balls.

- If both the bins are in L, then the algorithm picks the bins with the lesser load (fewer number of balls)
- If one of the bins is in L, then the algorithm chooses the other bin
- If both the bins are not in L, then the algorithm selects one of the bins randomly

The proof draws parallels from the 'always-go-left' algorithm mentioned in [1] and combines it with the proof for the power of two choices. The number of extra balls in a bin is the total number of balls on that bin minus T. It can be shown that the total number of extra balls can be upper bounded using the power of two choice proof by $O(\log\log n)$ and thus the maximum load in any bin is $T + O(\log\log n) = O\left(\frac{\delta \log n}{\log\log n}\right)$ balls *w.h.p.*

5 Conclusion

In this survey, we start by giving a brief introduction to the existing results for the power of choices problem. This is followed by a detailed summary of the technique used by [8] to show the $O(\log\log n)$ bounds with high probability. We then take a look at the problems involved in the real world implementation of the balls and bins problem including error and compression. The reported load may not always be equal to the actual load in the bin due to a host of reasons. In this survey, we also describe a technique in [7] which analyzes the bounds for the gap between the maximum load and the average load for the $(1 + \beta)$ choice case. The $(1 + \beta)$ choice can also be used to represent error in reporting load to the incoming balls. Finally, we take a look at the issues of compression with respect to limited memory and communication complexity. The paper by [2] proves the lower bounds for the maximum load in presence of compression case when we have $n^{1-\delta}$ bits of memory and communication. The authors then gives an algorithm with matching upper bounds when the system can utilize atmost $n^{1-\delta}$ bits for communication.

6 Future Work

The Power of Two Choices Problem can be used in a host of new applications such as improving the convergence of stochastic gradient descent and the coordinate descent algorithms. For example, in the stochastic gradient descent algorithm, we hope to show that if we use a power of d-choices type of technique and pick the sample with the highest gradient and try and proof the theoretical bounds for the convergence. The relationship between the maximum times any sample is selected and its effect on convergence can also be explored. The fundamental issue with implementing the power of two choices in the stochastic gradient descent algorithms is the potential advantage that might be obtained by using both the samples of the gradient instead of the maximum one.

Another area that can be explored is the designing of error tolerant algorithms and how analyse how it affects the performance of the power of two choices. For example in SGD, we can potentially use $(d-1)$ previous gradients and sampling only one new example per slot. But this introduces error in the gradient estimates and which will consequently stop us from using the traditional analysis of the power of choice problem. Finally the issue of compression of the state memory can be addressed where to keep track of the frequencies the database/system manager can maintain some sort of sketch of the frequency estimates of the actual system. This along with error guarantees in Power of Choices may provide theoretical guarantees on the effect of compression on load balancing.

References

[1] Yossi Azar, Andrei Z Broder, Anna R Karlin, and Eli Upfal. Balanced allocations. *SIAM journal on computing*, 29(1):180–200, 1999.

[2] Itai Benjamini, Yury Makarychev, et al. Balanced allocation: memory performance tradeoffs. *The Annals of Applied Probability*, 22(4):1642–1649, 2012.

[3] Petra Berenbrink, Artur Czumaj, Angelika Steger, and Berthold Vöcking. Balanced allocations: The heavily loaded case. *SIAM Journal on Computing*, 35(6):1350–1385, 2006.

[4] Yuval Peres, Kunal Talwar, and Udi Wieder. The (1+ β)-choice process and weighted balls-into-bins. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1613–1619. Society for Industrial and Applied Mathematics, 2010.

[5] Martin Raab and Angelika Steger. “balls into bins”—a simple and tight analysis. In *Randomization and Approximation Techniques in Computer Science*, pages 159–170. Springer, 1998.

[6] Andrea W Richa, M Mitzenmacher, and R Sitaraman. The power of two random choices: A survey of techniques and results. *Combinatorial Optimization*, 9:255–304, 2001.

[7] Kunal Talwar and Udi Wieder. Balanced allocations: the weighted case. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 256–265. ACM, 2007.

[8] Kunal Talwar and Udi Wieder. Balanced allocations: A simple proof for the heavily loaded case. In *Automata, Languages, and Programming*, pages 979–990. Springer, 2014.