Randomized Allocation Processes*

(EXTENDED ABSTRACT)

Artur Czumaj†
Volker Stemann‡

Abstract

We investigate various randomized processes allocating balls into bins that arise in applications in dynamic resource allocation and on-line load balancing. We consider the scenario when \( m \) balls arriving sequentially are to be allocated into \( n \) bins on-line and without using a global controller.

Traditionally, the main aim of allocation processes is to place the balls into bins to minimize the maximum load in bins. However, in many applications it is equally important to minimize the number of trails performed by the balls (the allocation time). We study adaptive allocation schemes that achieve optimal tradeoffs between the maximum load, the maximum allocation time, and the average allocation time.

We investigate allocation processes that may reallocate the balls. We provide a tight analysis of the maximum load of processes that during placing a new ball may reallocate the balls in up to \( d \) randomly chosen bins.

We study infinite processes, in which in each step a random ball is removed and a new ball is placed according to some scheduling rule. We present a novel approach that establishes a tight estimation of the time needed for the infinite process to be in the state near to its equilibrium.

Finally, we provide a tight analysis of the maximum load of the off-line process in which each ball may be placed into one of \( d \) randomly chosen bins. We apply this result to competitive analysis of on-line load balancing processes.

1 Introduction

Consider the following dynamic resource allocation problem. There is a system in which processes arrive one by one and each process has to choose on-line between a number of identical resources (for example, the process chooses a server to use among the servers in a network, or chooses a disk to store a directory). The system is not centralized and in order to test the state of the resource the process must submit a request to it. There are various parameters of such a system; the most important is the maximum load of the resource, another is the number of submitted requests. The dynamic resource allocation problem may be modeled as the problem of sequentially allocating balls into bins. In that case the balls correspond to the tasks allocated by the process and the bins correspond to the resources. The main aim of this paper is to analyze various strategies of placing balls into bins that arise in applications in dynamic resource allocation and on-line load balancing.

We investigate the problem of sequentially allocating \( m \) balls into \( n \) bins, \( m \geq n \). Different allocation rules may use different resources in various ways and may result in different distributions of the balls in the bins. In this paper we study allocation processes that allocate one ball after the other in a time-homogeneous (i.e., each ball is proceeded using the same protocol) and non-centralized way. Thus, for example, we exclude the deterministic scheme that places the first ball into the first bin, the second ball into the second bin, and so on, as this needs a global controller and is not time-homogeneous.

Within this class of allocation processes our main aim is to minimize the following parameters:

- the maximum load in any bin,
- the average allocation time of the balls, and
- the maximum allocation time of any ball.

Here the load of a bin is the number of balls in the bin and the allocation time is the number of bins chosen to candidate for the ball to be allocated.

One of the most effective ways to avoid global control of an allocation process is to use randomization. For instance, the very simple randomized process that puts each ball into a bin chosen i.u.r. (independently and uniformly at random) does not need a centralized controller, is time-homogeneous, its maximum and average allocation time is \( 1 \), and the maximum load is \( \Theta\left(\frac{\ln n}{\ln \ln n} + m/n\right) \), w.h.p.\(^1\) We shall refer to this process as the classical allocation process, CAP (see [14, 17] for a general exposition

\(^*\)Partially supported by EU ESPRIT Long Term Research Project 20244 (ALCOM-IT), by DFG-Sonderforschungsbereich 376 “Massive Parallelität: Algorithmen, Entwurfsmethoden, Anwendungen”, and by DFG Leibniz Grant Me872/6-1.

\(^†\)Department of Mathematics & Computer Science and Heinz Nixdorf Institute, University of Paderborn, Fürstenallee 11, D-33102 Paderborn, Germany. Email: artur@uni-paderborn.de. WWW: http://www.uni-paderborn.de/cs/artur.html.

\(^‡\)Research done while at the International Computer Science Institute.

\(^1\)Throughout the paper w.h.p. will denote that a given event holds with
of this process and its applications). Azar et al. [6] extended this process and showed that if each ball chooses i.u.r. with replacements $d > 1$ bins and then it is placed into the bin with the smallest load, then the maximum load decreases dramatically to $(1 + o(1)) \cdot \ln \ln n / \ln d + \Theta(m/n)$, w.h.p. We will refer to the process of Azar et al. as ABKU[d]. They also proved that each randomized on-line process that does not reallocate the balls and has the maximum allocation time $d$ must have a bin with load $(1 + o(1)) \cdot \ln \ln n / \ln d + \Omega(m/n)$, w.h.p. In the case $m = n$, Azar et al. [6] were able to tighten these bounds up to an additive constant term. They showed that the maximum load in ABKU[d] is $\ln \ln n / \ln d + \Theta(1)$ w.h.p. and that every randomized on-line process that has the maximum allocation time $d$ (and does not reallocate the balls) must have a bin with load $\ln \ln n / \ln d + \Omega(1)$ w.h.p.

Recently similar allocation schemes were analyzed in parallel and distributed setting. MacKenzie et al. [20] and Karp et al. [16] analyzed allocation schemes that appear in contention resolution protocols and shared memory simulations. Adler et al. [1] and Stemann [22] concerned with the tradeoff between the maximum load and the number of parallel communication rounds required to place all the balls. Berenbrink et al. [8] extended the approach of Stemann [22] to weighted balls.

Motivated by applications in dynamic load balancing Azar et al. [6] analyzed also infinite processes. There are $n$ balls that are initially allocated in $n$ bins in an arbitrary way. In each step a ball is chosen i.u.r. and removed from the system, and then a new ball is allocated according to the ABKU[d] rule: the ball is put into the least loaded among $d$ bins chosen i.u.r. with replacement. Azar et al. [6] showed that, independently of the initial distribution of the balls, after $O(n^3)$ steps the maximum load in any bin will be $\ln \ln n / \ln d + O(1)$ w.h.p. Another infinite model of randomized on-line load balancing, called the dynamic edge orientation problem, was studied by Ajtai et al. [2]. Mitzenmacher [21] analyzed a similar infinite and continuous randomized allocation process for the so-called supermarket model, in which the balls arrive as a Poisson stream and each nonempty bin removes the balls with exponential distribution with mean one.

All of the above allocation processes do not reallocate the balls during the run. It has been observed that the performance of many allocation processes may be significantly improved if the process can reallocate some of the balls (see, e.g., [5, 19, 23]). Notice that if the number of reallocations is not limited, it is trivial to maintain ideally balanced load. However, since the reassignments are expensive, their usage should be limited. Therefore in most of the existing load balancing algorithms the reassignments are usually performed in a very restricted way (see, e.g., [19, 23]).

### 2 New results

#### 2.1 Adaptive allocation processes

Although the ABKU[d] process significantly decreases the maximum load when compared to CAP, it increases the allocation time of a ball. The firm assumption about the allocation time of ABKU[d] implies, for example, that even if the first chosen bin is empty, another $d - 1$ bins are to be chosen for a given ball. Since in many applications an additional cost is charged for each access to a “bin”, this is a clear waste of resources. In order to avoid this additional cost we investigate adaptive allocation processes. In the adaptive process the number of choices made in order to place a ball depends on the loads of the bins previously chosen by the ball. We shall apply this approach to obtain an optimal tradeoff between the maximum load, the maximum allocation time, and the average allocation time.

We classify the adaptive allocation processes by an infinite nondecreasing sequence of positive integers $x_0, x_1, \ldots$; we allow some of $x_i$ be infinite. To each such a sequence we assign process $P(x_0, x_1, \ldots)$ modeled by the following algorithm:

- The $n$ balls are allocated sequentially.
- For each ball run the following protocol:
  - Let $M = 1$.
  - Repeat:
    - Choose a bin $b_M$ i.u.r. from $\{1, \ldots, n\}$.
    - Let $b$ be the bin of the minimum load among bins $\{b_1, b_2, \ldots, b_{M-1}\}$ and let $l$ be the load of $b$.
    - If $x_l \leq M$ then place the ball into bin $b$.
    - Else $M = M + 1$.
    - Until the ball is placed.

Note that the value of $x_l$ indicates that a ball can be placed into a bin with load $l$ after at least $x_l$ choices made by the ball. Observe also that process $P(x_0, x_1, \ldots)$ is time-homogeneous, in the sense that each ball is proceeded in the same way.

Notice that according to our discussion above it is reasonable to require that $x_0 = 1$. This corresponds to the situation where every time an empty bin is chosen, the ball is placed into it. We remark also that CAP is a $P(1, 1, \ldots)$ process and ABKU[d] is a $P(d, d, \ldots)$ process.

Our first result is a precise analysis of the maximum load of bins in $P(x_0, x_1, \ldots)$.

**Theorem 1** Let $n = m$ and let $x_0, x_1, \ldots$ be a fixed nondecreasing sequence of positive integers with $x_0 = \cdots = x_{s-1} = 1$ and $x_s > 1$. There exist positive constants $a$ and
\[ b \text{ such that if } (s - 1)! \leq \frac{a_n}{\log n}, \text{ then the maximum load in } P(s_0, s_1, \ldots) \text{ is} \]
\[ \min \{ i \geq s : ((s - 1)!/b)n^{a_i} > n \} + \Theta(1) \text{ w.h.p.} \]

**Theorem 2** Let \( \alpha = m/n \geq 1 \) and let \( x_0, x_1, \ldots \) be a fixed nondecreasing sequence of positive integers. Let \( \alpha = \min \{ i : 12/e \cdot \alpha^{i+1} \leq i \} \) and \( b = \max \{ i : (\frac{3}{e})^{i} \cdot \alpha^{i} \leq 6 \ln n \} \). Suppose that \( x_b > 1 \) and let \( s = \min \{ i \geq a : x_i > 1 \} \). Then the maximum load of \( P(s_0, s_1, \ldots) \) is
\[ \min \{ i \geq s : (s!/(4\alpha^{i+1}))n^{a_i} > m \} + \Theta(1) \text{ w.h.p.} \]

Using Theorem 2 one can verify, for example, that if \( x_3 > 1 \) and \( n = m \), then the maximum load is \( k + \Theta(1) \) w.h.p., where \( k \) is the smallest integer such that \( \prod_{j=3}^{k} x_j > \ln n \). Hence this bound implies also the bound for the maximum load of ABKU(d) given in [6].

The main difficulty in the proofs of Theorems 1 and 2 is caused by the fact that (unlike in the analysis of [6]) the number of bins chosen for a ball may vary and by the situation when \( x_{s-1} = 1 < x_s \) and \( s = \omega(1) \). Our proofs are a combination of the standard analysis of CAP and the analysis presented in [6] coupled with some new ideas. Since the proofs of Theorems 1 and 2 are rather technical, they are omitted in this extended abstract.

### 2.2 Allocation time of adaptive processes

Most of the previous work concerned with the issue of minimization of the maximum load. In certain applications it is desirable to not only minimize the maximum load of the bins, but also other parameters of the system. One main reason to study the adaptive strategies is that they may achieve tradeoff between different performance measures. We study adaptive schemes that achieve optimal tradeoffs between the maximum load, the maximum, and the average allocation time.

We first analyze a very simple \( \text{THRESHOLD}(m, n, M) \) process, which is the adaptive process \( P(s_0, s_1, \ldots) \) that places \( m \) balls into \( n \) bins with \( x_0 = x_1 = \cdots = x_{M-1} = 1 \) and \( x_M = +\infty \). That is, in order to place a ball we choose one by one bins i.i.d. with replacement until a bin with load less than \( M \) is selected; then the ball is put into this bin. We provide tight upper and lower bounds for the average allocation time for \( \text{THRESHOLD} \). In particular, we show that already the very simple process \( \text{THRESHOLD}(n, n, 2) \) performs very well when only the maximum load and the average allocation time are concerned.

**Theorem 3** \( \text{THRESHOLD}(n, n, 2) \) has the maximum load at most 2 (with certainty) and the average allocation time at most \( 1.146194 + o(1) \), w.h.p.

Actually, one can show that this tradeoff between the maximum load and the average allocation time is optimal (among all on-line random allocation processes) up to lower-order terms.

The only disadvantage of \( \text{THRESHOLD}(n, n, 2) \) is that its maximum allocation time is \( \Theta(\ln n) \) w.h.p. We can obtain the full tradeoff between the maximum load, the maximum allocation time, and the average allocation time if we couple \( \text{THRESHOLD} \) with ABKU(d), and investigate adaptive processes \( P(s_0, s_1, \ldots) \) with \( x_0 = x_1 = \cdots = x_{M-1} = 1 \) and \( x_M = x_{M+1} = \cdots = d \).

**Theorem 4** Let \( n = m \) and \( d > 1 \) be integer.

1. Process \( P_{(1, 1, d, d, \ldots)} \) (that is, process \( P(s_0, s_1, \ldots) \) with \( x_0 = x_1 = 1 \) and \( x_i = d \) for \( i \geq 2 \)) has the maximum load \( \ln n / \ln d + \Theta(1) \), the maximum allocation time at most \( d \), and the average allocation time at most \( 1.146194 + o(1) \), w.h.p.

2. For every constant \( \epsilon > 0 \) and for sufficiently large \( n \) there exists an adaptive allocation process that has the maximum load \( \ln n / \ln d + \Theta(1) \), the maximum allocation time at most \( d \), and the average allocation time at most \( 1 + \epsilon \), w.h.p.

3. If \( d = \ln^{\Omega(1)} n \) then there exists an adaptive allocation process that has the maximum load \( (1 + o(1)) \cdot \ln n / \ln d \), the maximum allocation time at most \( d \), and the average allocation time \( 1 + o(1) \), w.h.p.

Since every process with the maximum allocation time at most \( d \) must have the maximum load \( \ln n / \ln d + \Omega(1) \) w.h.p. [6], these processes achieve the best possible tradeoffs between the maximum load, the maximum allocation time, and the average allocation time.

We can also provide some extensions of Theorems 3 and 4, and analyze the case \( m \neq n \).

### 2.3 Applications of adaptive processes

Our analysis of adaptive processes leads to many interesting applications. We present here one that extend results obtained by Azar et al. [6].

**Dynamic Resource Allocation:** Consider a non-centralized system in which a user has to place on-line a task in one of identical and non-distinguishable servers (or to store a file on a disk, etc.). The user may check the load of all servers and put the task into the least loaded server. Although this process leads to ideally balanced distribution of the tasks, it is expensive, since it requires sending a message to each server in the system and interrupting its work. The ABKU[2] scheme can be used to obtain a more efficient solution. In a system with \( n \) tasks and \( n \) servers, if each user samples the load of two resources chosen i.i.d. and submits the task to the least loaded one, then the total overhead caused by communication with the
servers is $2n$, and the load of the $n$ servers varies by only $\ln \ln n / \ln 2 + \Theta(1)$ w.h.p.

Using adaptive processes we can design more efficient schemes. For example, the user may sample the load of resources chosen i.u.r. until it finds a server with at most one task and then submit the task to that server. Then the maximum load is 2 and Theorem 3 implies that the total number of messages sent is at most $(1.146194 + o(1))n$ w.h.p.

If one wants to keep small also the number of messages sent for each task, then we could use the adaptive processes from Theorem 4. For example, let us choose arbitrarily $d \geq 2$ and a positive constant $\varepsilon$. If each user samples the resources according to the adaptive process from Theorem 4 (2), then the total number of messages sent is not larger than $(1 + \varepsilon)n$, each user sends at most $d$ messages, and the load of the $n$ servers varies by at most $\ln \ln n / \ln d + O(1)$ w.h.p. Notice, for example, that if $d$ is chosen such that $d \geq \ln^c n$ for some positive constant $c$, then the difference in the loads is constant, w.h.p.

Our approach that uses Theorem 4 can be also applied (after some small modifications) to more realistic systems in which the number of tasks differs from the number of servers and/or when the number of tasks is not known in advance.

2.4 Reassignments

One of the main applications of the ABKU[d] scheme is in on-line load balancing. The balls may be viewed as non-distinguishable tasks that are arriving sequentially and are to be assigned to the servers (corresponding to the bins) in a distributed system. In many applications it is possible that some of the existing tasks (balls) may be reassigned. As it turns out, the performance of many load balancing algorithms may be significantly improved if we allow reassignments of the balls (see, e.g., [5, 19, 23]). If the number of reassignments is not limited, it is trivial to maintain ideally balanced load. However, since the reassignments are usually expensive, we would like to perform only a limited number of reassignments and made them only locally.

We investigate how much can be gained by using allocation processes with reassignments. We consider a scenario where during the allocation of a new ball one may ask for some number (called the allocation time) of possible locations and then arbitrarily reassigns the balls in the chosen bins. The lower bound of $\ln \ln n / \ln d + \Omega(1)$ for the maximum load of any process with maximum allocation time $d$ due to Azar et al. [6] does not hold for processes with reallocations. We study the question whether allocation processes with reassignments may substantially decrease the maximum load.

We begin with the analysis of a simple modification of ABKU[d], which we call BAL[d]: upon arrival of a ball choose $d$ bins i.u.r. with replacement, assign the ball to any of the chosen bins, and then balance the load of all the bins chosen. Here, by balancing we mean that the total load of the chosen bins remains unchanged and the loads of any two of the chosen bins differ by at most one. We provide a precise analysis of the maximum load of BAL[d].

**Theorem 5** Let $d \geq 2$. The maximum load of BAL[d] is $\Theta(\ln \ln n / \ln d + m/n)$, w.h.p. For $m = n$, the maximum load of BAL[d] is $\ln \ln n / \ln d + \Theta(1)$, w.h.p.

Actually, our lower bound holds for a much larger class of allocation processes that may perform reassignments. We prove a tight lower bound that holds for all on-line allocation processes that may perform reassignments.

**Theorem 6** Let $A$ be any (on-line) allocation process that assigns $m$ balls to $n$ bins according to the following scheme: Upon arriving of a ball $A$ chooses i.u.r. with replacement up to $d$ bins, assigns the ball to any of the chosen bins, and arbitrarily reassigns the balls among the chosen bins using complete information about the distribution of the current load of all bins in the system. Then the maximum load of any bin is $\Omega(\ln \ln n / \ln d + m/n)$, w.h.p. For $m = n$ the maximum load of any bin is $\ln \ln n / \ln d + \Omega(1)$, w.h.p.

This theorem shows that even thought it is possible to design an allocation process (e.g., BAL[d]) that reassigns the balls and has the maximum load smaller than any process that does not allow reassignments, the benefit of using reallocations is very small. The ball reallocations do not help too much, since ABKU[d] achieves optimal maximum load (up to an additive constant term in the case $n = m$) even if reassignments of the balls are allowed.

2.5 Infinite allocation processes

We study also infinite allocation processes: there are initially $m$ balls arbitrarily allocated in $n$ bins and in each round a ball chosen i.u.r. is removed and then a new ball is placed according to some scheduling rule. We consider the infinite versions of ABKU[d] and of BAL[d], in which a new ball added in the infinite process is placed according to the scheduling rule of (the finite) ABKU[d] and of (the finite) BAL[d], respectively.

We can combine our analysis of BAL[d] with the analysis of the infinite version of ABKU[d] due to [6] to show that after sufficiently many rounds the maximum load of the infinite version of BAL[d] will be $\ln \ln n / \ln d + O(1)$, w.h.p. Then we study the rate of convergence of infinite processes to their stationary distribution. We present a simple and general technique that uses coupling (cf. [3, 18]) to obtain accurate bounds for the rate of convergence to the stationary distribution of the maximum loads of various infinite allocation processes. In particular, we improve
the bounds for the rate of convergence of ABKU[d] due to Azar et al. [6] and prove the following theorem.

**Theorem 7** Let \( n \) balls be arbitrarily allocated in \( n \) bins and let \( d \geq 2 \). Then, independently of the initial distribution of the balls in the bins, after \((1 + o(1)) \cdot n \ln n\) rounds of the infinite version of ABKU[d] the maximum load is \(\ln \ln n / \ln d + O(1)\), w.h.p.

Observe that this bound for the rate of convergence is tight up to lower-order terms. Indeed, it is easy to construct initial allocation of the balls such that after \((1 - o(1)) \cdot n \ln n\) rounds of the infinite version of ABKU[d] the maximum load will be \(\omega(\ln \ln n / \ln d)\) w.h.p.

We can also obtain the same bounds for the infinite version of BAL[d].

**Theorem 8** Let \( n \) balls be arbitrarily allocated in \( n \) bins and let \( d \geq 2 \). Then, independently of the initial distribution of the balls in the bins, after \((1 + o(1)) \cdot n \ln n\) rounds of the infinite version of BAL[d] the maximum load is \(\ln \ln n / \ln d + O(1)\), w.h.p.

Our technique is fairly general and can be easily extended to provide tight bounds also for the case \( m \neq n \). Details are omitted here.

### 2.6 Competitive on-line load balancing and off-line allocations

In the competitive analysis of on-line load balancing processes it is important to study the behavior of optimal off-line solutions. Let us suppose that all random choices made by ABKU[2] are given in advance. In that case, when the full information about the random choices is known, Azar et al. [6, Lemma 12] showed that one can assign the \( n \) balls into the \( n \) bins with the maximum load \( O(1) \) w.h.p. We strengthen their result and prove the following theorem.

**Theorem 9** Suppose that \( 2n \) random choices in \( n \) pairs performed as we would run ABKU[2] are given in advance. We require that ball \( i \) is placed into one of the bins in the \( i \)th pair. Then one can place the \( n \) balls into \( n \) bins so that the maximum load will be 2, w.h.p.

The proof of the theorem is based on the analysis of random multigraphs. One can also easily prove that this result is optimal, in the sense that for \( d = o(\ln n) \) no assignment with maximum load 1 exists w.h.p.

Theorem 9 immediately tightens competitive analysis of the on-line load balancing process given in [6, Section 5.1.1]. Theorem 14 in [6] deals with scheduling permanent tasks against distribution \( P_d \) (see [6] for the definition of distribution \( P_d \) and for more details). Azar et al. showed that the ABKU[d] algorithm achieves the competitive ratio \( O(\ln \ln n / \ln d) \), w.h.p., and no algorithm can do better. Our bound strengthens this result to the competitive ratio \( \ln \ln n / 2 \ln d + O(1) \), w.h.p., which is tight up to an additive constant term.

### 2.7 Organization of the paper

In Section 3 we study the average allocation time of adaptive processes and outline the proofs of Theorems 3 and 4. In Section 4 we analyze allocation processes with reassignments and sketch the proofs of Theorems 5 and 6. Section 5 deals with infinite allocation processes and provides the proof of Theorem 7. Finally, in Section 6 we analyze off-line allocation processes and outline the proof of Theorem 9.

In this extended abstract all proofs are either sketched or omitted. They are deferred to the full paper.

### 3 Minimizing the average allocation time in adaptive processes

In this section we outline the proofs of Theorems 3 and 4. Our results are based on the analysis of process \( THRESHOLD(m, n, M) \).

It is not very difficult to prove that for \( M \geq 2 \) the average allocation time of \( THRESHOLD(n, n, M) \) is constant w.h.p. Our aim here is to provide an accurate estimation how big is this constant. The following main theorem gives a tight (but implicit) bound.

**Theorem 10** Let \( M > \alpha = m/n, m = \omega(\sqrt{n \ln n}) \), and let \( \zeta \) be the solution of

\[
\alpha = \zeta \sum_{k=0}^{M-2} \frac{e^k}{k!} + M \left(1 - \sum_{k=0}^{M-1} \frac{e^k}{k!}\right).
\]

The average allocation time of \( THRESHOLD(m, n, M) \) is at least \( \zeta - o(1) \) and at most \( \zeta + o(1) \), w.h.p.

**Sketch of the proof:** For a given non-negative integer \( x \) let \( \psi_M(x) = \min \{M, x\} \) and for given non-negative integers \( x_1, \ldots, x_n \) let \( \Psi_M(x_1, \ldots, x_n) = \sum_{i=1}^{n} \psi_M(x_i) \).

Let \( X_1, X_2, \ldots \) be the infinite sequence of independently and uniformly distributed random variables corresponding to the trials made in \( THRESHOLD(m, n, M) \), i.e., if the \( i \)th trial returns bin \( j \) then \( X_i = j \). Denote by \( Y \) the number of trials required to place all \( m \) balls. Let \( \lambda_i(t) \) be the number of balls in bin \( i \) at time \( t \) in \( THRESHOLD(m, n, M) \). For a given sequence \( X_1, X_2, \ldots \), let \( \lambda_i^*(\tau) \) denote the load of bin \( j \) after \( \tau \) trials if we would run (an infinite) CAP with the sequence of bin choices \( X_1, X_2, \ldots \). That is, \( \lambda_i^*(\tau) = |\{ j : 1 \leq j \leq \tau \text{ & } X_j = i \}| \). The crucial observation we use here is that if \( t = \Psi_M(\lambda_1^*(\tau), \ldots, \lambda_M^*(\tau)) \), then \( \lambda_i(t) = \psi_M(\lambda_i^*(\tau)) \) for each \( i \). Thus \( Y \) is equal to the smallest \( \tau^* \) such that \( m = \Psi_M(\lambda_1^*(\tau^*), \ldots, \lambda_M^*(\tau^*)) \). Therefore in order to find the value of \( \tau^* \) it is sufficient to analyze process CAP.
For a given $T$ we estimate the probability that $T \geq \tau^*$ and $T \leq \tau^*$ using the Poisson approximation heuristic (see, e.g., [1, 4, 7]). We approximate the behavior of $\Psi_M(\lambda_1^*(T), \ldots, \lambda_n^*(T))$ by function $\Psi_M$ taken on $n$ independent random variables with Poisson distribution, each with parameter $T/n$. We obtain the bounds in the theorem by estimating the values of $T$ for which $\Psi_M(\lambda_1^*(T), \ldots, \lambda_n^*(T))$ is larger or respectively smaller than $m = an$ w.h.p.

Unfortunately, there is no any closed form for $\zeta$ defined by the formula in Theorem 10, even in the case $m = n$. The only easy case is when $M = 1$ (and hence we require $m < n$), and then we obtain $\zeta = \ln(n/(n-m))$. Nevertheless, for $M = 2$ we can show that $\zeta = -W_{-1}((\alpha - 2) \cdot e^{-2}) - 2$, where $W$ is the Lambert function (cf. [10]). Therefore by applying the asymptotic bound for $W_{-1}(e^{-2})$ we can derive Theorem 3.

In order to prove Theorem 4 we first state the following consequence of Theorem 10.

**Lemma 3.1** Let $\alpha > \varepsilon > 0$. There exists a constant $c > 0$ such that if $M > \alpha \geq 1$ is even, $M \geq 8$, and $M$ satisfies $M \geq c \cdot \frac{\ln((1+\alpha)/e)}{\ln(1+\alpha/e)}$, then the average allocation time of $\text{THRESHOLD}^\varepsilon(n, n, M)$ is at most $\alpha + \varepsilon + O(1/(M \cdot \sqrt{n \ln n}))$, w.h.p. □

**Corollary 3.2** (1) For any positive constant $\varepsilon$ there exists a constant $M$ such that for sufficiently large $n$ the maximum allocation time of $\text{THRESHOLD}(n, n, M)$ is at most $1 + \varepsilon$, w.h.p. (2) If $M = \omega(1)$ then the average allocation time of $\text{THRESHOLD}(n, n, M)$ is $1 + o(1)$, w.h.p. □

**Proof of Theorem 4**:

Let us fix $M$ and consider adaptive process $P(x_0, x_1, \ldots)$ with $x_0 = x_1 = \cdots = x_{M-1} = 1$, $x_M = x_{M+1} = \cdots = d$. It is obvious that the maximum allocation time of any ball in $P(1, 1, d, \ldots)$ is at most $d$. Using Theorem 2 we see that if $M$ is a constant then the maximum load of $P(x_0, x_1, \ldots)$ is $\ln n / \ln d + O(1)$ w.h.p. For $M = \omega(1)$, $M = o(\ln n / \ln d)$, and $M = o(\ln n)$ the maximum load of $P(x_0, x_1, \ldots)$ is $(1 + o(1)) \cdot \ln n / \ln d$ w.h.p. The crucial observation we use now is that the average allocation time of any ball in $P(x_0, x_1, \ldots)$ is stochastically dominated by the average allocation time of any ball in $\text{THRESHOLD}(n, n, M)$. Hence we combine our argument above with Theorem 3 and Corollary 3.2 to complete the proof. □

**4 Reassignments**

In this section we sketch the proofs of Theorems 5 and 6. We first show that the maximum load of any bin in $\text{ABKU}[d]$ stochastically dominates the maximum load of any bin in $\text{BAL}[d]$. Then we present a general lower bound for processes that may reallocate the balls. This together will yield Theorems 5 and 6.

Fix $n$. For an $n$-vector $v = (v_1, \ldots, v_n) \geq 0$ let $\pi_v$ be a permutation of $n$ elements such that $v_{\pi_1}(1) \geq v_{\pi_2}(2) \geq \cdots \geq v_{\pi_n}(n)$. If $v_2 \geq v_2 \geq \cdots \geq v_n$ (i.e., we may choose $\pi_v$ as the identity permutation) then we say $v$ is normalized. The operation of applying $\pi_v$ to $v$ so that the resulting vector $v^\ast$ is normalized is called normalization of $v$. Let $e_i$ be the $n$-vector $e_i = (e_{i,1}, e_{i,2}, \ldots, e_{i,n})$ such that $e_{i,j} = \delta_{i,j}$.

Let $v$ and $u$ be two $n$-vectors vectors and let $v^\circ = (v_1^\circ, \ldots, v_n^\circ)$ and $u^\circ = (u_1^\circ, \ldots, u_n^\circ)$ be normalizations of $v$ and $u$, respectively. We say that $v$ majorizes $u$, which is denoted by $v \succcurlyeq u$, if $\sum_{j=1}^i v_j^\circ \geq \sum_{j=1}^i u_j^\circ$ for every $i$, $1 \leq i \leq n$. Observe, in particular, that if $v \succcurlyeq u$ then $\max_{1 \leq i \leq n} v_i \geq \max_{1 \leq i \leq n} u_i$.

**Lemma 4.1** The maximum load of any bin in $\text{ABKU}[d]$ stochastically dominates the maximum load of any bin in $\text{BAL}[d]$.

**Proof**: We present the proof only for $d = 2$; the proof for larger $d$ is analogous. The proof is based on the coupling technique and it is similar to the proof of Theorem 7 of Azar et al. [6].

We first extend Lemma 6 from [6] and show the following proposition (whose proof is omitted).

**Proposition 4.2** Let $v = (v_1, \ldots, v_n)$ and $u = (u_1, \ldots, u_n)$ be two normalized integer $n$-vectors. Let $i$ and $j$ be two arbitrary integers, $1 \leq i < j \leq n$. If $u_i > u_j + 2k - 1$ for some $k \geq 1$, then $v_i + e_j \geq u - ke_i + (k + 1)e_j$.

Let $\lambda_i(t)$ be the load of bin $i$ after time $t$ in $\text{ABKU}[2]$ and let $\lambda_i^*(t)$ be the load of bin $i$ after time $t$ in $\text{BAL}[2]$. Let $\lambda(t)$ and $\lambda^*(t)$ be the load vectors of $\text{ABKU}[2]$ and $\text{BAL}[2]$, respectively, at time $t$. That is, $\lambda(t) = (\lambda_1(t), \lambda_2(t), \ldots, \lambda_n(t))$ and $\lambda^*(t) = (\lambda_1^*(t), \lambda_2^*(t), \ldots, \lambda_n^*(t))$.

Fix $t$. Let $\Omega_t$ be the set of all vectors of length $2t$ of elements in $\{1, \ldots, n\}$. Any vector $\omega = (\omega_1, \omega_2, \ldots, \omega_{2t}) \in \Omega_t$ determines uniquely the $2t$ choices made by $t$ balls, so that bins $\omega_{2t-1}$ and $\omega_{2t}$ were chosen by ball $r$. Our aim now is to show that there exists a 1-1 function $f_t : \Omega_t \rightarrow \Omega_t$ such that if $\omega$ represents the choices made in $\text{ABKU}[2]$ and $f_t(\omega)$ represents the choices made in $\text{BAL}[2]$, then the load vector of $\text{ABKU}[2]$ majorizes the load vector of $\text{BAL}[2]$. This would complete the proof.

Our construction of $f_t$ is by induction on $t$. For $t = 0$ we can take the identity function. Now assume that we have such a function $f_t$ and we show how to construct function $f_{t+1}$. $f_{t+1}$ restricted to the first $2t$ elements is equal to
Thus we only have to determine \( f_{i+1} \) on the last two elements.

Fix \( \omega = (\omega_1, \ldots, \omega_{2t}) \). Let \( \lambda(t) \) be the load vector of ABKU[2] for bins \( \omega \) and let \( \lambda^*(t) \) be the load vector of BAL[2] for bins \( f_t(\omega) \). To simplify the notation, let us assume that \( \lambda_1(t) \geq \cdots \geq \lambda_{2t}(t) \). Let \( \pi = \pi_{\lambda^*(t)} \) be a permutation such that \( \lambda^*_\pi(1, t) \geq \cdots \geq \lambda^*_\pi(2t, t) \). We set \( f_{t+1} \) so that to restrict the last two elements, \( f_{t+1} \) maps \((\omega_{2t+1}, \omega_{2t+2})\) into \((\pi(\omega_{2t+1}), \pi(\omega_{2t+2}))\). What we have to show now is that \( \lambda(t) \geq \lambda^*(t) \) implies \( \lambda(t+1) \geq \lambda^*(t+1) \). Here \( \lambda(t+1) \) and \( \lambda^*(t+1) \) are the load vectors of ABKU[2] and BAL[2] for bins \( \omega^* = (\omega_1, \ldots, \omega_{2t+2}) \) and \( f_{t+1}(\omega^*) \), respectively.

Assume, without loss of generality, that \( i \leq j \). Then \( \lambda(t+1) = \lambda(t) + \varepsilon_j \). Because \( i \leq j \), we obtain that \( \lambda_{\pi(1)}(t) - \lambda_{\pi(1)}(t) = r_2 \geq 0 \). Hence

\[
\lambda^*(t+1) = \lambda^*(t) + \left( \frac{r}{2} \right) \cdot e_{\pi(1)} + \left( \frac{r}{2} \right) + 1 \cdot e_{\pi(j)}.
\]

Now we can apply Proposition 4.2 to complete the proof.

This lemma combined with the analysis of ABKU[d] due to Azar et al. [6] immediately implies the upper bound for the maximum load of BAL[d] in Theorem 5. The lower bound follows from Theorem 6 that is proven below.

**Sketch of the proof of Theorem 6**

Assume first that \( m = n \) and let, without loss of generality, \( d = o((\ln n)) \). Without restriction of generality we can assume that \( A \) chooses exactly \( d \) bins for each ball. Let us define recursively \( \gamma_0 = n \) and

\[
\gamma_{i+1} = \frac{1}{2} \cdot n \cdot 2^{-(i+1)} \cdot \left( \frac{\gamma_i}{2^{1+d/\ln n}} \right)
\]

We say we are at time \( t \) if we consider the situation after placing \( t \) balls. Let \( B_t \) be the set of bins with load at least \( i \) at time \( n(1 - 2^{-i}) \). Denote by \( F_t \) the event that \( |B_t| \geq \gamma_t \). Let \( L_t = \{ t \in T : n(1 - 2^{-(i+1)}) < t \leq n(1 - 2^{-i}) \} \).

Denote by \( L_i \) the set of bins that are chosen exactly once during the \( d \cdot |L_t| \) trials of balls from \( T_t \). Let \( C_{i+1} = B_t \cap L_{i+1} \).

Suppose that \( F_t \) holds, that is, after \( n(1 - 2^{-i}) \) steps at least \( \gamma_i \) bins have load at least \( i \). Let us consider the bins in \( C_{i+1} \). Each such a bin is chosen exactly once during the interval-time \( L_{i+1} \) and therefore it changes its load in \( L_{i+1} \) at most once. Suppose that a ball \( t \) in \( L_{i+1} \) chooses \( d \) bins from \( C_{i+1} \). Since the load of each chosen bin is at least \( i \), at least one of the chosen bin will have load at least \( i + 1 \) at time \( t + 1 \). In that case let one of the chosen bins with load at least \( i + 1 \) at time \( t + 1 \) be called the representative. Since the representative is chosen only once in \( L_{i+1} \), it will belong to \( B_{i+1} \). Let \( D_{i+1} \) be the set of representatives in the interval-time \( L_{i+1} \). Thus \( D_{i+1} \subseteq B_{i+1} \). Our aim is to estimate a lower bound for the size of \( D_{i+1} \).

We can show that if we set \( t^* \) equal to the largest \( i \) that satisfies

\[
\gamma_i \geq \frac{n \cdot \ln n \cdot e^{d/2}}{d} \quad \text{and} \quad \gamma_{i+1} \geq \sqrt{d \cdot n \cdot \ln n / 2^i}.
\]

Then \( t^* = \ln \ln n = \Theta(1) \). Moreover, for every \( i < t^* \), conditioned on \( F_{t^*} \), \( |D_{i+1}| \geq \gamma_{i+1} \) holds w.h.p. Since (by our initial setting of \( \gamma_0 \)) \( F_0 \) holds w.h.p., we obtain that \( F_{t^*} \) holds w.h.p. as well. Now one can calculate that, conditioned on \( F_{t^*} \), \( |B_{i+1}| \geq n^{3/4} \). The last \( n/2^{t^*} \) balls perform \( d n/2^{t^*} = o(n \ln n) \) random choices. Hence standard calculations can be used to show that with high probability at least one element from \( B_{t^*} \) will be chosen by any of the last \( n/2^{t^*} \) balls. This completes the proof of the case \( n = m \).

Now let us consider the case \( m > n \). Since the theorem is obvious for \( m/n = \Omega(\ln \ln n / \ln d) \) and \( d = \Omega(\sqrt{\ln n}) \), we only must show that if \( m = O(n \ln n / \ln d) \) and \( d = o(\sqrt{\ln n}) \) then the maximum load is \( \Omega(\ln n / \ln d) \) w.h.p. Using similar arguments to those given above we obtain that for \( t^* = \ln \ln n = \Theta(1) \) with high probability holds \( |B_{t^*+1}| \geq n^{3/4} \). The last \( n - (n(1 - 2^{-t^*})) \) balls will make \( O(md) = o(n \ln n) \) trials. Therefore one can easily show that the last \( m - (n(1 - 2^{-t^*})) \) balls will not choose all the bins in \( B_{t^*+1} \) w.h.p., which completes the proof of Theorem 6.

**5 Infinite processes**

In this section we investigate infinite allocation processes and sketch the proof of Theorem 7. We focus here only on the case \( m = n \).

The main idea of the proof is as follows. Let us consider a Markov chain \( M \) on integers whose state at time \( t \) corresponds to the maximum load at round \( t \) of the infinite ABKU[d]. Azar et al. [6] proved that after \( O(n^2) \) rounds of the infinite ABKU[d] the maximum load is \( \ln n + O(1) \), w.h.p., independently of the initial distribution of the balls. This immediately implies that in the stationary distribution \( \pi_M \) of \( M \) the maximum load is \( \ln n + O(1) \), w.h.p. Once we know this property of \( \pi_M \), we may study the rate of convergence of \( M \) to \( \pi_M \). We apply the coupling technique and tighten the bound of Azar et al. [6] by showing that after \((1 + o(1)) \cdot n \ln n \) steps the maximum load of \( M \) will have almost the stationary distribution. This will imply Theorem 7.

It is worth pointing out that our approach differs significantly from that of Azar et al. [6]. Azar et al. first provided a bound for the maximum load of the infinite ABKU[d] and from that they concluded properties of \( \pi_M \). We follow a different approach and analyze the rate of convergence to \( \pi_M \) without using properties of \( \pi_M \). Only at the very end we combine the result of Azar et al. [6] about \( \pi_M \) to prove Theorem 7. (We also emphasize that we do not provide an alternative proof that the maximum load after some number
of rounds of the infinite ABKU[d] is $\ln \ln n / \ln d + O(1)$, w.h.p.)

5.1 Basics

For any random variable $X$ its probability distribution is denoted by $\mathcal{L}(X)$. For any two random variables $X$ and $Y$ defined jointly on the same space, we shall measure the separation between their probability distributions by *variation distance* between $\mathcal{L}(X)$ and $\mathcal{L}(Y)$ which is defined as $\|\mathcal{L}(X) - \mathcal{L}(Y)\| = \sup_{A} |\Pr[X \in A] - \Pr[Y \in A]|$.

The main technique used in our analysis of the rate of convergence of infinite allocation processes is *coupling* (see, e.g., [3, 18] for a more detailed exposition).

**Definition 5.1** Consider a Markov chain $\mathcal{M} = (\mathcal{M}_t)_{t \in \mathbb{N}}$ with state space $\mathcal{X}$. A *coupling* $(X_t, Y_t)_{t \in \mathbb{N}}$ for $\mathcal{M}$ is a Markov chain on $\mathcal{X} \times \mathcal{X}$ such that

$$\mathcal{L}(X_t) = \mathcal{L}(\mathcal{M}_t), \mathcal{L}(Y_t) = \mathcal{L}(\mathcal{M}_0)$$

and

$$\mathcal{L}(Y_t) = \mathcal{L}(\mathcal{M}_t), \mathcal{L}(Y_0) = \mathcal{L}(\mathcal{M}_0).$$

We shall use the *coupling inequality* (see, e.g., [3]).

**Lemma 5.1** Let $\mathcal{M}$ be an ergodic Markov chain with state space $\mathcal{X}$ and let $\pi$ be the stationary distribution of $\mathcal{M}$. Let $(X_t, Y_t)_{t \in \mathbb{N}}$ be a coupling for $\mathcal{M}$. Let $\mathcal{T}$ be a random time (called the coupling time) such that $X_t = Y_t$ for all $t \geq \mathcal{T}$ and any $\mathcal{L}(Y_0)$. Then

$$\|\mathcal{L}(X_t) - \pi\| \leq \Pr[\mathcal{T} > t].$$

Thus, if we can find a coupling, we get an immediate bound on $\|\mathcal{L}(X_t) - \pi\|$ in terms of the tail probabilities of the coupling time $\mathcal{T}$.

For an $n$-vector $v = (v_1, \ldots, v_n)$ we use the standard notation $\|v\|_1 = \sum_{i=1}^n v_i$. For every $i \in \{1, \ldots, n\}$, we write $v \oplus e_i$ to denote the normalized vector (cf. Section 4) of $v + e_i$. Similarly, $v \oplus e_i^n$ denotes the normalized vector of $v - e_i$. We denote by $\Phi_n$ the set of all normalized $n$-vectors $v$ satisfying $\|v\|_1 = n$.

We model the state space of the infinite ABKU[d] by normalized $n$-vectors $v \in \Phi_n$. We say we are at state $v$ if there exists a permutation $\sigma$ such that $v_i$ represents the current load of bin $\pi(i)$. Vector $v$ will be called the *load vector*. The infinite ABKU[d] process is modeled by Markov chain $(\mathcal{M}_t)_{t \in \mathbb{N}}$ on state space $\Phi_n$. Transitions of $(\mathcal{M}_t)_{t \in \mathbb{N}}$ are governed by the following rule:

- Let $v = \mathcal{M}_t$.
- Choose a number $i \in \{1, \ldots, n\}$ independently at random according to the probability distribution $\mathcal{P}(v)$ such that $\Pr(\mathcal{P}(v)[i = j] = \frac{v_j}{\|v\|_1}$ for every $j \in \{1, \ldots, n\}$.
- Let $v^* = v \oplus e_i$.
- Choose a number $t \in \{1, \ldots, n\}$ independently at random according to the probability distribution $\mathcal{P}(v)$ such that $\Pr(\mathcal{P}(v)[t = j] = \frac{(n+1-i)j - (n-j)j}{n^2} \forall j \in \{1, \ldots, n\}$.
- Set $\mathcal{M}_{t+1} = v^* \oplus e_t$.

One can easily verify that $(\mathcal{M}_t)_{t \in \mathbb{N}}$ behaves properly and that $\mathcal{L}(\mathcal{M}_t)$ equals the probability distribution of the load vector after $t$ rounds of the infinite ABKU[d]. Let $\Pi$ be the stationary distribution of $(\mathcal{M}_t)_{t \in \mathbb{N}}$. (One can easily show that $\mathcal{P}(\mathcal{M}_t)$ has a unique stationary distribution $\Pi$.) Our aim is to show that for some $t = (1 + o(1)) \cdot n \ln n$ we have $\|\mathcal{M}_t - \Pi\| = 1 - o(1)$, which will yield Theorem 7.

5.2 Coupling

We define coupling $(v(t), u(t))_{t \in \mathbb{N}}$ for $(\mathcal{M}_t)_{t \in \mathbb{N}}$. Below we describe one transition in the coupling. Full coupling $(v(t), u(t))_{t \in \mathbb{N}}$ is defined such that $(v(t + 1), u(t + 1)) = (v(t)^o, u(t)^o)$ for any $t \in \mathbb{N}$.

Let $v, u \in \Phi_n$. Let $w = v - u, w^+ = (w_1^+, \ldots, w_n^+)$, and $w^- = (w_1^-, \ldots, w_n^-)$, where $w_i^+ = \max\{w_i, 0\}$ and $w_i^- = \min\{w_i, 0\}$. Let also $z = (z_1, \ldots, z_n)$ such that $z_i = \min\{v_i, u_i\}$. Define also $\Delta(w) = \|w^+\|_1 = \sum_{i=1}^n \max\{w_i, 0\}$.

**One-step-Coupling** $(v, u)$:

1. Choose a number $T \in \{1, \ldots, n\}$ i.i.r.

   - If $T \leq \|z\|_1$ then choose $s$ at random according to the probability distribution $\mathcal{P}(v)$ and set
     - $v^* = v \oplus e_s$, $u^* = u \oplus e_s$.
   - If $T > \|z\|_1$ then choose at random $s$ according to the probability distribution $\mathcal{P}(w^-)$ and $t$ according to the probability distribution $\mathcal{P}(w^+)$, and set
     - $v^* = v \ominus e_s$, $u^* = u \ominus e_t$.

2. Choose a number $r$ according to the distribution $\mathcal{R}$. Set

   - $v^o = v^* \oplus e_r$, $u^o = u^* \oplus e_r$.

3. Return $(v^o, u^o)$.

**Lemma 5.2** Let $w^o = v^* - u^*$ and $w^o = v^o - u^o$. One-step-Coupling $(v, u)$ satisfies the following properties:

- One-step-Coupling $(v, u)$ properly defines one step of $(v(t), u(t))_{t \in \mathbb{N}}$ (cf. Definition 5.1).
- $\|v^o\|_1 \leq \Delta(w^*) \leq \Delta(w)$.
- $\|v^o\|_1 \leq \Delta(w^*)$.

Now let us consider the full coupling. Let $w(t) = v(t) - u(t)$ and $\Delta(w(t)) = 1 - o(1)$. Therefore (by Lemma 5.1) our aim is to analyze the random variable $\mathcal{T}$ that is the minimum $t_0$ such that
\Delta(\mathbf{w}(\tau)) = 0 \text{ for all } \mathbf{v}(0), \mathbf{u}(0), \tau \geq t_0.

Properties of the coupling given in Lemma 5.2 imply that \( \Delta(\mathbf{w}(t)) \) is non-increasing with \( t \) and that if \( T > \|\mathbf{z}(t)\|_1 \) in step \( t \), then \( \Delta(\mathbf{w}(t+1)) \leq \Delta(\mathbf{w}(t)) - 1 \). Observe that \( \|\mathbf{z}(t)\|_1 = n - \Delta(\mathbf{w}(t)) \) and that \( \Delta(\mathbf{w}(t)) \leq n - 1 \). Let \( G_t \) be random variable with the geometric distribution with parameter \( a/n \). One can show that \( T \) is stochastically dominated by \( G^* = \sum_{i=1}^{n} G_i \). Thus we can apply standard arguments to show that for any \( \gamma = \gamma(n) \):

\[
\Pr[\|\mathbf{z} \cdot (\ln n + \gamma)\|_1 \leq e^{-\gamma}.
\]

Hence, by the coupling inequality (Lemma 5.1), we obtain for \( t = n(\ln n + \gamma) \)

\[
\|G_t - \Pi\|_1 \leq e^{-\gamma}.
\]

Let \( A \) be the subset of load vectors in \( \Phi_\infty \) that have the maximum load of at most \( \ln n / \ln d + \mathcal{O}(1) \). That is,

\[
A = \{ \mathbf{v} \in \Phi_\infty : v_1 \leq \ln n / \ln d + \mathcal{O}(1) \}.
\]

Azar et al. [6] proved that \( \Pi(A) = 1 - o(1) \). On the other hand, inequality (1) implies that for \( t = n(\ln n + \gamma) \)

\[
\|\Pr[M_t \in A] - \Pi(A)\|_1 \leq e^{-\gamma}.
\]

Hence \( \Pr[M_t \in A] \geq 1 - e^{-\gamma} - o(1) \) for \( t = n(\ln n + \gamma) \), which completes the proof of Theorem 7. \( \square \)

6 Off-line allocations

In this section we prove Theorem 9. Our proof is based on the analysis of random multigraphs studied previously by Janson et al. [13] and (implicitly) by Karp et al. [16]. Generate i.u.r. with replacement \( n \) ordered pairs \( (x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \), \( 1 \leq x_i, y_i \leq n \). Now let us define a random multigraph \( M \) that has \( n \) vertices \( \{1, \ldots, n\} \) and \( n \) undirected edges \( \{ (x_i, y_i) : 1 \leq i \leq n \} \). Notice that \( M \) may contain self-loops and parallel edges. We prove the following key lemma.

Lemma 6.1 Let \( M \) be a random multigraph with \( n \) vertices and \( n \) edges. Then one may orient the edges of \( M \) so that each vertex has out-degree at most 2, w.h.p.

Sketch of the proof: We first study edge-orientation of acyclic and unicyclic multigraphs.

Let \( G \) be an isolated tree. We can orient all the edges of \( G \) so that each vertex will have out-degree 1: choose an arbitrary vertex as the root and then direct all the edges towards the root. Notice that the out-degree of the root is 0.

If \( G \) is a unicyclic multigraph (connected multigraph with the same number of vertices and edges) then we may perform a similar construction. Let \( T \) be an arbitrary spanning tree of \( G \) and let \( (x, y) \) be the unique edge of \( G \) that does not belong to \( T \). Choose \( x \) as the root of \( T \) and perform the orientation of the edges of \( T \) so that each edge is directed towards the root. Because \( x \) has out-degree 0 in \( T \), we can orient \( (x, y) \) to \( y \).

Our crucial observation is that \( M \) possesses almost the same properties as a random graph in the \( G_{n, m} \) model [9, 11, 15], that is, a random simple graph with \( n \) vertices and \( n \) edges chosen i.u.r. without replacement. In particular, the classical result of Erdős and Rényi [11] (see also [9, Theorem VI.11] and [15, Theorem 3.3]) implies that \( M \) consists w.h.p. of one ”giant” connected component, unicyclic connected components, isolated trees, and isolated vertices.

Let \( C \) denote the giant component, let \( \nu \) be the number of vertices in \( C \), and let \( \kappa \) denote the number of edges of \( C \). It is easy to see that \( C \) is a random connected multigraph with \( \nu \) vertices and \( \kappa \) edges. One can show that \( \nu = (a + o(1))n > .7967n \) and hence \( \kappa = (\nu - 1) \leq .2033n \). w.h.p., where \( a \) is the only positive solution of the equation \( e^{-2a} + a = 1 \). We can show that since \( \kappa/\nu \leq 1.2552 \) w.h.p., \( C \) can be partitioned (w.h.p.) into two edge-disjoint graphs \( G_1 \) and \( G_2 \) such that \( G_1 \) is a spanning tree of \( C \) and \( G_2 \) is a sum of unicyclic components, isolated trees, and isolated vertices.

Thus we can decompose \( G \) (w.h.p.) into two edge-disjoint graphs \( G_1 \) and \( G_2 \). \( G_1 \) consists of \( G_1 \) and unicyclic connected components, isolated trees, and isolated vertices outside \( C \). \( G_2 \) consists of \( C \) and \( G_2 \). Therefore we may apply edge-orientation of acyclic and unicyclic multigraphs to orient all the edges in \( G_1 \) so that each vertex will have out-degree at most 1. Similarly we can orient the edges of \( G_2 \). This gives us orientation of the edges of \( G \) in which each vertex has out-degree at most 2 w.h.p. \( \square \)

Proof of Theorem 9:

One can model random choices performed in the ABKU[2] process by a multigraph \( M \) with \( n \) vertices and \( n \) edges. The vertex set of \( M \) corresponds to the set of bins. If \( (x, y) \) is the \( i \)th chosen pair of bins, then \( (x, y) \) is the \( i \)th undirected edge of \( M \). Observe that since the bins are generated i.u.r., the obtained multigraph \( M \) is random according to our definition above. Now we use Lemma 6.1 to orient the edges of \( M \), so that, w.h.p., each vertex has out-degree at most 2. We assign the balls to the bins using this orientation. If the \( i \)th edge \( (x, y) \) is directed from \( x \) to \( y \), then we place ball \( i \) into bin \( x \). Because there are at most two edges out-going from \( x \), the load of bin \( x \) is at most 2. \( \square \)

References


