J. Parallel Distrib. Comput. 74 (2014) 2065–2076

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Balls into non-uniform bins

Petra Berenbrink [a], André Brinkmann [b], Tom Friedetzky [c], Lars Nagel [b,*]

[a] School of Computing Science, Simon Fraser University, Burnaby, B.C., V5A 1S6, Canada
[b] Zentrum für Datenverarbeitung, Johannes-Gutenberg-Universität Mainz, 55099 Mainz, Germany
[c] School of Engineering and Computing Sciences, Durham University, Durham DH1 3LE, United Kingdom

HIGHLIGHTS

- This paper investigates randomised multiple-choice balls-into-bins games.
- Such a balls-into-bins game models the allocation of tasks to servers of different speeds/capacities.
- It is shown that a balanced allocation can be achieved if the number of balls equals the total capacity.
- Simulations of other cases suggest that our algorithm works even for a small amount of balls and bins.

ARTICLE INFO

Article history:
Received 15 December 2012
Received in revised form
17 October 2013
Accepted 31 October 2013
Available online 7 November 2013

Keywords:
Load balancing
Randomised algorithms
Balls-into-bins games

ABSTRACT

Balls-into-bins games for uniform bins are widely used to model randomised load balancing strategies. Recently, balls-into-bins games have been analysed under the assumption that the selection probabilities for bins are not uniformly distributed. These new models are motivated by properties of many peer-to-peer (P2P) networks. In this paper we consider scenarios in which non-uniform selection probabilities help to balance the load among the bins. While previous evaluations try to find strategies for identical bins, we investigate heterogeneous bins where the "capacities" of the bins might differ significantly. We look at the allocation of m balls into n bins of total capacity C where each ball has d random bin choices. For such heterogeneous environments we show that the maximum load remains bounded by $\ln\ln(n)/\ln(d)+\mathcal{O}(1)$ *w.h.p.* if the number of balls m equals the total capacity C. Further analytical and simulative results show better bounds and values for the maximum loads in special cases.

1. Introduction

In the standard balls-into-bins game, m unit-sized balls are allocated to n identical bins. It is assumed that every ball independently and uniformly at random chooses d bins and that it commits itself to a least-loaded of these bins. The goal of this strategy is to balance the load among the bins, by minimising the maximum number of balls allocated to any bin.

Balls-into-bins games are successfully used to model randomised load balancing strategies in networks and many other "real world" applications (see, e.g., [17,18,10,19,15] for applications of randomisation strategies). In these cases, balls represent requests or data items, while bins model servers or some form of storage. Most of the previous papers assume the same uniform capacity (or size) for all bins and uniform bin probabilities. The goal is to balance the load in a way that each bin receives approximately the same number of balls. We should point out that when we refer to a bin's "capacity"/"size" then we do not mean to imply the existence of a maximum "volume", or load threshold (as in e.g. bin packing); the reader should think more in terms of "speed", "bandwidth" or "compression ratio". The precise notion that we use throughout this paper is simply that when a ball of size s is placed into a bin of capacity c, then the "effective" load that this bin experiences is $\ell = s/c$.

Standard balls-into-bins games assume that the probability of a bin to be selected by a ball is the same for all bins. It is unfortunately very difficult to maintain this property in distributed environments without centralised control. P2P environments like Chord or CAN [18,17], e.g., are unable to map peers evenly to their address space, making some bins more likely to be selected than others [13,7,5]. Byers et al. [7,8] extended the model, assuming that the probability for a bin to be selected within a random experiment is not uniform over all bins. Their underlying process still tries to balance the number of balls as evenly as possible over the set of bins.

However, in many practical applications, some bins can handle a much larger load than others, under-utilising stronger bins under these constraints. In the variant of balls-into-bins games that we consider, it is assumed that the bins are not uniform, but that they come with an integer capacity, as outlined above.

* Corresponding author.
E-mail addresses: petra@cs.sfu.ca (P. Berenbrink), brinkman@uni-mainz.de (A. Brinkmann), tom.friedetzky@dur.ac.uk (T. Friedetzky), nagell@uni-mainz.de, larsn25@yahoo.de (L. Nagel).

0743-7315/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jpdc.2013.10.008

Let the total capacity C be the sum of the capacities of all bins. The natural probability for a bin to be chosen would be either $1/n$, that is uniform, or c_i/C, proportional to the bin's capacity c_i. We will analyse the latter case for $d \geq 2$ and show that the maximum load is $\ln\ln(n)/\ln(d) + \mathcal{O}(1)$ *w.h.p.* (Theorem 3). Furthermore, we will investigate cases in which the maximum load is constant (Theorems 1 and 2). In some cases changing the probability distribution leads to much better results (Theorem 5).

1.1. Related work

There is a vast number of papers dealing with balls-into-bins games in their many different settings. We restrict our attention to major results and previous work that is relevant to the results presented in this paper.

In the standard game in which each ball chooses d bins *i.u.r.*, the maximum load can be bounded by $\ln\ln(n)/\ln(d) + \Theta(1)$ if $m = n$ balls are thrown [2]. In case $m \gg n$ the deviation of the load from the average m/n is also $\ln\ln(n)/\ln(d) + \Theta(1)$ and thus independent of the number of balls m [4].

Recently, several papers have examined the case in which bins are not chosen *i.u.r.* The motivation for these models comes from the properties of peer-to-peer networks like Chord, which use Consistent Hashing to distribute requests (balls) over computers (bins) [13,18]. There the computers and requests are mapped to random points on a ring and the requests are assigned to the closest computer on the ring in an anti-clockwise direction. Therefore, each bin is responsible for one arc on the ring. The maximum arc length can be up to a factor of $\log(n)$ larger than the average arc length.

Byers et al. [7,8] successfully apply the power-of-two-choices paradigm to this setting by letting each request randomly choose $d \geq 2$ points and allocate itself to a peer of lowest load. Although the maximum arc length can be up to $\log(n)$ times larger than the average one, the maximum load of every peer is still bounded by $\ln\ln(n)/\ln(d) + \Theta(1)$, *w.h.p.*, for $m = n$. Hence, the work of Byers et al. shows that this imbalance does not lead to a shift in the maximum load for the case $m = n$.

Wieder [21] demonstrates that in the scenario of Byers et al. the maximum difference between the loads grows with m. Thus, for $m \gg n$ the bounds are not as tight as in the standard case [4]. However, if the number of choices d is allowed to (slowly) grow with the deviation in the probability distribution, the maximum load is again bounded by $\frac{m}{n} + \mathcal{O}(\ln\ln(n))$ (which complies with [4]). The presented bounds are tight in a way that a smaller d leads to a deviation of the load linear in m.

Kenthapadi and Panigrahy [14] and Godfrey [11] analyse graph-based models in which the random choices are non-uniform and dependent. In [14] the 2-choice game is considered with the restriction that balls can only choose bins that are connected by an edge in an underlying graph G. The authors assume that each edge in the graph has the same probability to be chosen and show that the maximum load does not deviate much from the maximum load in the standard 2-choice game provided that G is (almost) n^ϵ-regular where ϵ is a large enough constant. In [11] Godfrey generalises the model to the d-choice game by assuming that the underlying graph is a d-uniform multi-hypergraph (which is even allowed to change with each ball). A ball can only choose sets of d bins that correspond to hyperedges in its hypergraph. Again, each hyperedge has the same probability of being chosen. Godfrey investigates under which circumstances the maximum load in any bin is 1 *w.h.p.* The authors of [3] consider the same model and improve the results of [11].

The case of heterogeneous bin sizes has been considered in the related field of selfish load balancing (e.g., [9,1]), but to our knowledge nobody has analysed it for multiple-choice games. Such games are mentioned by Wieder [21] to motivate his work about multiple-choice games with heterogeneous probabilities. He suggests to choose the bins' probabilities proportional to their capacities. In this paper we will analyse this particular case and variations of it.

1.2. Our contributions

All previous results assume that each bin has a uniform capacity and that the balls should be distributed as evenly as possible. In contrast, we assume that the system consists of heterogeneous bins where each bin i can have an arbitrary, integral capacity c_i and the objective is to balance the load of each bin, which is defined as the number of balls inside this bin divided by its capacity. If not stated otherwise, we assume that a bin's probability of being chosen is proportional to its size.

In the analytical part of this paper, we assume that we have n bins with a total capacity of C and $m = C$ balls. Hence, the optimal maximum load is one. The main analytical result from Section 3 shows that the maximum load of a bin is $\ln\ln(n)/\ln(d) + \mathcal{O}(1)$ (Theorem 3) if $d \geq 2$. Hence, the maximum load does not grow with an increasing capacity. We even show that the maximum load becomes constant if almost all bins are big, *i.e.*, have size $\Omega(\ln(n))$ (Theorem 1). Provided that we can choose a different probability distribution, a constant maximum load can be achieved even if there is only a constant fraction of $\Theta(\ln\ln(n))$-sized bins (Theorem 5). The proof of this theorem uses Observation 2, which states that if all bins have the same capacity $\bar{c}$, the maximum load is bounded by $(m/n + \mathcal{O}(\ln\ln(n)))/\bar{c}$ *w.h.p.*

Based on a simulation environment, we arrange and simulate bin arrays with varying parameters in Section 4 and compare our analytical results with the experiments. There we also consider settings that we do not analyse, most notably the heavily loaded case and systems with a small number of bins.

2. Model and definitions

We assume bins to be non-uniform. Each bin comes with a positive integer capacity which we also refer to as size. We denote the capacities/sizes of the n bins by $c_1, \ldots, c_n$, and let $C = \sum_{i=1}^{n} c_i$.

We usually allocate $m = C$ balls into our system of n bins and assume that each ball has $d \geq 2$ choices. In the following we say that a bin is *chosen* or that a ball *chooses* a bin if we refer to the d choices of a ball. When a bin actually receives the ball, then we say that the bin *gets* or *is allocated* the ball.

The load balancing protocol (see Algorithm 1) greedily tries to minimise the maximum load within the set B of the d chosen bins after a ball has been allocated. It therefore determines the subset $B_{opt} \subseteq B$ with the smallest load after a possible allocation and *i.u.r.* allocates the ball to one of the bins with the highest capacity from it. We will show within the analysis that it is beneficial to move the load into the direction of these bigger bins.

We say that if m_i balls are allocated to a bin b_i of capacity $c_i \geq 1$, then this bin's load is $\ell_i = \frac{m_i}{c_i}$. Usually we will assume that the probability of bin b_i with capacity c_i being chosen is $\frac{c_i}{C}$ and therefore proportional to c_i. If we use other probability distributions, we will clearly point this out.

Algorithm 1 Load Balancing Protocol

1: **for all** balls **do**
2: Independently choose a set B of d bins at random
3: Determine the set $B_{opt} \subseteq B$ of bins that would have the lowest load after allocating the ball
4: Determine the maximum capacity $c_{\max}$ of the bins in B_{opt}
5: Remove all bins b_j with capacity $c_j < c_{\max}$ from B_{opt}
6: *i.u.r.* choose a bin from B_{opt} and allocate the ball to it
7: **end for**