Random dropping policies simulation

Francesco Pagano
s299266

1 Introduction

In the current laboratory activity the aim was simulating different policies for random dropping to analyze if any of the tested policies was able to achieve the optimal occupancy level of 1 ball per bin taking track of $O(1)$ bins; in particular the following algorithms were tested:

- Vanilla random dropping: drop a ball in a randomly selected bin without any consideration regarding its occupancy level
- Random dropping with load balancing: drop a ball in a bin selected among d randomly picked bins with the lowest occupancy level. In particular, $d = 2$ and $d = 4$ were tested.

2 The simulation model

In the delivered simulation, Balls and Bins are instances of their respective classes. Each ball has two attributes: a unique index and the index of the bin it has been dropped into; each bin has its own unique index and the number of balls dropped inside. Of course each ball can be dropped into one and only one bin, but each bin can contain as many balls as necessary. Independently of the employed policy, the decision process over the selection of a bin is governed by an instance of a random variable uniformly distributed. The **input parameters** for this simulation are:

- Number of bins and balls, both equal to N
- D: the amount of bins to be selected in the first steps of load balancing.

The **output** of the simulation consists of a graph showing the occupancy level of the three policies in function of an increasing number of bins(balls). As far as the minimum occupancy is concerned, it is always 0, while the average occupancy is always 1 due to the fact that number_bins = number_balls. To keep track of Bin and Ball classes' instances and the data regarding the maximum and minimum occupancy level for the different policies, simple python lists were employed and some log files to better store and visualize data.

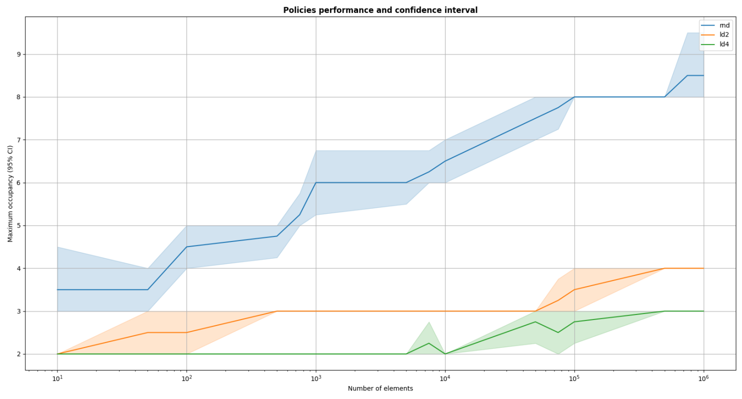

Figure 1: Occupancy levels for the three tested policies with a 95% confidence interval.

3 Comments

As shown in Fig.1, the three tested policies feature very different behaviours in terms of optimizing the occupation level for each bin. Theoretically, the vanilla random policy should be upper bounder by:

$$max\_occupancy = 3\frac{\log n}{\log\log n} \tag{1}$$

while load balancing has its maximum occupancy sharply concentrated around:

$$max\_occupancy = \frac{\log\log n}{\log d} + O(1) \tag{2}$$

Since in this test case the maximum number of items in the simulation is 10^6, then the vanilla random dropping policy should be around 5.26, while for the load balanced policy the maximum occupancy should be around 3.78 for $d = 2$ and 1.89 for $d = 4$. As Fig.1 shows these theoretical constrains are respected within a 95% confidence interval.