# A New Mechanism of Object Random Picker

Maoji-Programming

September 28, 2020

## 1 Introduction

For most people, choosing the canteen for lunch is the biggest problem in our daily life. The general solution is a random picker to help users to decide where they go. However, there are some weaknesses in this solution.

First, it causes continuously repeated selection, which means that the picker gets the same result as the previous drawing. Users may feel tedious when go to the restaurant again. To prevent this situation, the weight list will be updated manually which is a troublesome job.

Second, Editing the weight of all selection is not flexible when using the general random pickers. After users add a new choice, they need to turn the weight value and make sure that sum of the weight equals 1. Also, when users delete a choice, not only they need to turn the weights, but also keep the ratio of the weight. In most cases, they arbitrarily update the weights that seems reasonable. The lack of update principles is the problem that makes the picking probability unfair.

## 2 Objective

We need a random picker with a mechanism that:

1. automatically distributes the weights properly after adding, deleting and picking a choice
2. the weight of the choice result is set to 0.
3. automatically distributes the weights based on the rating after picking a choice

# 3 Implementation

## 3.1 Data structure

The original data list only contains records' name and weight. In relational schema, name of the choice is the primary key.

$$List(\overline{name}, weight)$$

For original picking mechanism, the appearing frequency is proportional to weight in general. To fulfill the second principle of the mechanism, the weight is always changed such that the appearing frequency is not proportional to weight of the choice. we add a new attribute 'rating' that affects the appearing frequency of the further picking. In relational schema, name of the choice is the primary key.

$$List'(\overline{name}, weight, rating)$$

After the weight of picking result is set to 0, according to the third principle, this weight will be distributed to other choice based on their rating, the higher rating it is, the more weight it gets. We define the range of rating is **1** to **5**, **1** is the lowest and **5** is the highest.

To make sure that the mechanism works properly, we allow creation for a empty record list that size can be 0. We assume the size of the record list is larger than 0 when operating deletion. For Selection, the minimum size of the record list is 2

### 3.2 Algorithm

**Algorithm 1:** weight distribution mechanism after adding a selection

**Function** `create(`$\hat{n}$, $L$`):`

**input:**

$\hat{n}$, the name of the new object that will be inserted into list $L$

$L$, the object list that attributes $n, w, r$ each records represents as name, weight and rating of the choice

**procedure:**

#Updating original records

**foreach** *record* $(n, w, r)$ ***in*** $L$ **do**

  updated record($n'$,$w'$,$r'$) = ($n$,$w \times \frac{|L(n,w,r)|}{|L(n,w,r)|+1}$,$r$)

**end**

#delete record

$\hat{L} \leftarrow \{(\hat{n}, \frac{1}{|L(n,w,r)|+1}, 3)\} \cup L$

**return** 1

**Theorem 1.** *The mechanism of creation works properly in anytime adding new records.*

*Proof.* The invariant of the weight updating mechanism is for all sizes of list $N > 0$, the sum of weight still equal to 1.

When list contain one choice with weight = 1, after the new record is inserted into the list, the weight of the new choice = $\frac{1}{1+1} = 0.5$, the original weight = $\frac{1}{1+1} = 0.5$ the proposition holds in base case N = 1.

Assume a List $L$ with weight $(w_1, w_2, ..., w_N)$and size N is an arbitrary number. Given that $\sum_{i=1}^{N} w_i = 1$. When the new record is insert into the list, the weight of new record will be $\frac{1}{N+1}$ and the weight of $i$-th original records = $\frac{Nw_i}{N+1}$

$$\begin{aligned}
\text{sum of the weight} &= \frac{Nw_1}{N+1} + \frac{Nw_2}{N+1} + ... + \frac{Nw_n}{N+1} + \frac{1}{N+1} \\
&= \frac{N}{N+1}(\sum_{i=1}^{N} w_i) + \frac{1}{N+1} \\
&= \frac{N}{N+1}(1) + \frac{1}{N+1} \\
&= 1
\end{aligned}$$

the proposition holds in for all positive integer size of the record list. □

**Algorithm 2:** weight distribution mechanism after deleting a selection

**Function** `delete(`$\hat{n}$, $L$`)`**:**

**input:**

$\hat{n}$, the name of the object that will be deleted into list $L$

$L$, the object list that attributes $n, w, r$ each records represents as name, weight and rating of the choice

**procedure:**

#Updating remaining records

**foreach** *record* $(n, w, r)$ ***in*** $L$ **do**

  updated record($n'$,$w'$,$r'$) = ($n$,$w + \frac{\hat{w}}{|L|-1}$,$r$)

**end**

#deleting the record

$\hat{L} \leftarrow L \backslash \{(\hat{n},\hat{w},\hat{r})\}$

**return** 1

**Theorem 2.** *The mechanism of deletion works properly in anytime removing new records.*

*Proof.* The invariant of the weight updating mechanism is for all size of list N $> 1$, the sum of the weight still equal to 1. When the list contains two choices with weights $w, 1-w$ given that sum of all weights equal 1.

When first choice deleted,the sum of the weight $= 1 - w + \frac{w}{2-1} = 1$. In the base case, the proposition holds.

Suppose there is a list L with $N+1$ choices $\{w_1, w_2, ...w_N, w_d\}$ given that sum of all weights equal 1. , when a choice with weight $w_d$ deleted, the weight of updated choices $= \{w_1 + \frac{w_d}{N}, w_2 + \frac{w_d}{N}, ..., w_N + \frac{w_d}{N}\}$, the sum of the updated weight

$$\begin{aligned} &= w_1 + w_2 + ... + w_N + \frac{w_d \times N}{N} \\ &= w_1 + w_2 + ... + w_N + w_d \\ &= 1 \end{aligned}$$

the proposition holds in for all positive integer size $> 1$ of the record list. □

**Algorithm 3:** weight distribution mechanism after picking a selection

**Function** `draw(`$L$`)`**:**

**input:**

$L$, the object list that attributes $n, w, r$ each records represents as name, weight and rating of the choice

**procedure:**

#Picking one

result = pick(L)

#Updating records

$r_m = mean(L \setminus \{result\})$

**foreach** *record* $(n, w, r)$ ***in*** $L$ **do**

  **if** $(n, w, r)$ == *result* **then**

    updated record($n'$,$w'$,$r'$) = $(n, 0, r)$

  **else**

    updated record($n'$,$w'$,$r'$) = $(n, w \times \frac{w_{result}}{|L|-1} \times \frac{r}{r_m}, r)$

  **end**

**end**

**return** result

The weight update mechanism make the weight always changed, when we take one draw, the probability of getting a item is dependence to the weight. When we take draws $n$ times, the frequency of getting the items is independence the weight of the items in round $i \in [1, n]$. Therefore, we add a new parameter rating denoted as $r$, to control the frequency of the record choice.

Then, the update mechanism for the weight of not selected items $w$:

$$w' \leftarrow w \times \frac{w_{result}}{|L| - 1} \times \frac{r}{r_m}$$

$\frac{w_{result}}{|L|-1}$ represents the average weight division of drawn item.
$\frac{r}{r_m}$ represents the rating of the item compared with the average.

if we abandon the rating system, the $\frac{r}{r_m}$ can be set to 1. In general, the initial weight of each item is $\frac{1}{N}$

**Theorem 3.** *The mechanism of drawing works properly in anytime picking a record.*

*Proof.* When $n = 2$ that is in base case, after the one item $(n_1, 1 - w, r_1)$ drawn, the weight another one $w$ becomes $w + (1 - w) \times \frac{1}{1} \times \frac{r_2}{r_2} = 1$.
the proposition holds. Suppose the list is $\{(n_1, w_1, r_1), (n_2, w_2, r_2), ..., (n_N, w_N, r_N)\}$, the sum of the weight equals 1.
After drawing a item $(n_i, w_i, r_i)$, the updated list becomes $\{(n_1, w_1 + \frac{w_i}{N-1} \times \frac{r_1}{r_m}, r_1), (n_2, w_2 + \frac{w_i}{N-1} \times$

$\frac{r_2}{r_m}, r_2), ..., (n_i, 0, r_i), ..., (n_N, w_N + \frac{w_i}{N-1} \times \frac{r_N}{r_m}, r_N)\}$ Sum of the weight

$$
\begin{aligned}
&= w_1 + \frac{w_i}{N-1} \times \frac{r_1}{r_m} + w_2 + \frac{w_i}{N-1} \times \frac{r_2}{r_m} + ... + 0 + ... + w_N + \frac{w_i}{N-1} \times \frac{r_N}{r_m} \\
&= w_1 + w_i \times \frac{r_1}{\sum_{j=[1,n] \cap j \neq i} r_j} + w_2 + w_i \times \frac{r_2}{\sum_{j=[1,n] \cap j \neq i} r_j} + ... + 0 + ... + w_N + w_i \times \frac{r_N}{\sum_{j=[1,n] \cap j \neq i} r_j} \\
&= w_1 + w_2 + ... + w_i \times \frac{\sum_{j=[1,n] \cap j \neq i} r_j}{\sum_{j=[1,n] \cap j \neq i} r_j} + ... + w_N \\
&= 1
\end{aligned}
$$

The proposition always holds that select a item, the sum of the weight still equals 1. □

**Theorem 4.** *Without rating mechanism or supposed the rating of each choices are the same, there are N choices in the list,* $\boldsymbol{E}[Pr(i,n)] = \frac{1}{N}$

*Proof.* Initially, the weight of each item equals $\frac{1}{N}$ in a list with size $N$. Probability of getting i-th item in round n $Pr[i,n] =$

$$
\begin{cases}
0, \text{if } i \text{ drawn in round } n-1 \\
Pr[i, n-1] + Pr[j, n-1] \times \frac{1}{N-1}
\end{cases}
$$

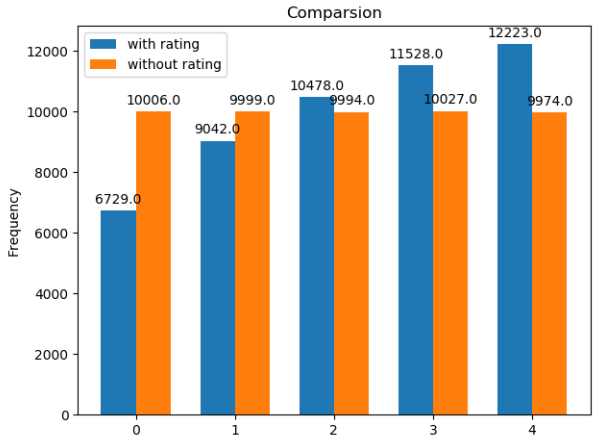
it takes $Pr[i, n-1]$ chance will get 0 probability in selecting i$^{th}$-item in round $n$. And it takes $1 - Pr[i, n-1]$ chance will get $Pr[i, n-1] + Pr[j, n-1] \times \frac{1}{N-1}$ in selecting i$^{th}$-item in round $n$.

$$
\begin{aligned}
\mathbf{E}[Pr(i,1)] &= (\frac{1}{N} + \frac{1}{N}\frac{1}{N-1})(1 - \frac{1}{N}) \\
&= \frac{1}{N}(1 + \frac{1}{N-1})(1 - \frac{1}{N}) \\
&= \frac{1}{N} \\
\mathbf{E}[Pr(i,2)] &= (1 - \mathbf{E}[Pr[i,1]])(\mathbf{E}[Pr[i,1]] + \frac{\mathbf{E}[Pr[j,1]]}{N-1}) \\
&= (1 - \frac{1}{N})(\frac{1}{N}(1 + \frac{1}{N-1})) \\
&= \frac{1}{N} \\
\mathbf{E}[Pr(i,n)] &= (1 - Pr[i,n-1])(Pr[i,n-1] + \frac{Pr[j,n-1]}{N-1}) \\
&= \frac{1}{N}
\end{aligned}
$$

□

We simulate 50000 drawing in 3 choices. The graph shows the result that without the rating mechanism, the choices of list are uniformly distributed. We still cannot find the expected probability

of choice with the rating system. However, the graph show that the rating can affect the frequency (expected probability) of the choices




**Algorithm 4:** a straightforward drawing algorithm with continuous duplication

**Function** `draw(`$L$`)`**:**

**input:**

$L$, the object list that attributes $n, w, r$ each records represents as name, weight and rating of the choice

**procedure:**

#Picking one

result = pick(L)

#Check if it is fulfill the condition

**while** *result == prev_result* **do**

  result = pick(L)

**end**

prev_result = result

**return** result

Compare with Algorithm 4 which is also a random picker promising getting non-duplicated results, Algorithm 3 have a little superiority.

1. From the perspective of time complexity, they required constant time. Algorithm 3 always requires 1 round but Algorithm 4 perform $O(\frac{N}{N-1})$ rounds which high-probably requires 1 round.

2. From the perspective of space complexity, they required constant space. Algorithm 3 requires no extra memory space to store any data but Algorithm 4 need 1 integer space to store the previous result.

## References