Alias method

Source:
www.keithschwarz.com/darts-dice-coins/

Balaji Lakshminarayanan

Feb 19, 2015

Why alias method?

You are given an n-sided die where side i has probability p_i of being rolled.

How do you efficiently simulate S rolls of the die (for large S)?

Why alias method?

You are given an n-sided die where side i has probability p_i of being rolled.

How do you efficiently simulate S rolls of the die (for large S)?

- Vanilla inverse CDF method: $O(nS)$
  - each sample costs $O(n)$

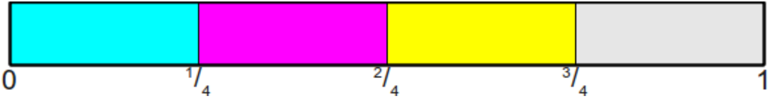
Why alias method?

You are given an n-sided die where side i has probability p_i of being rolled.

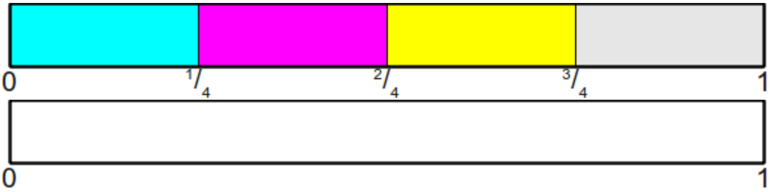
How do you efficiently simulate S rolls of the die (for large S)?

- Vanilla inverse CDF method: $O(nS)$
  - each sample costs $O(n)$
- Vose's alias method: $O(n)$ pre-processing + $O(S)$
  - each sample costs $O(1)$!

Simulating a fair die

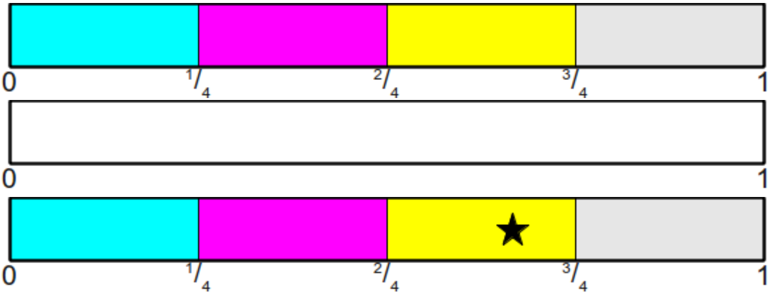


Simulating a fair die



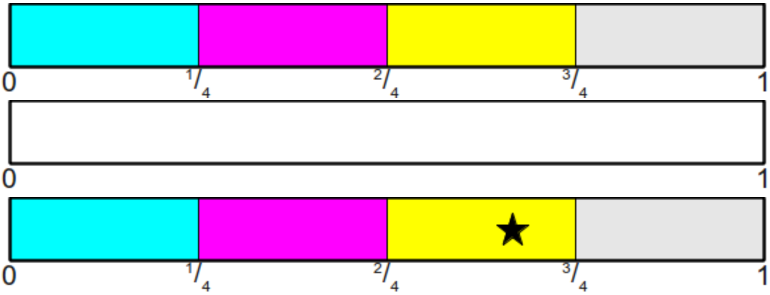
1. Sample x uniformly from $[0, 1)$

Simulating a fair die



1. Sample x uniformly from $[0, 1)$
2. Return floor(xn)

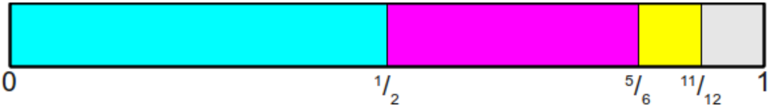
Simulating a fair die



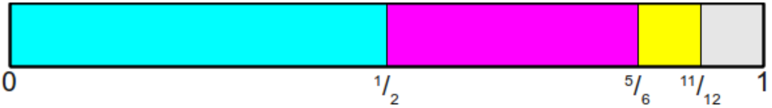
1. Sample x uniformly from $[0, 1)$
2. Return floor(xn)

- Generate uniform r. v. in $O(1)$
- Find bin in $O(1)$

Simulating a loaded die using inverse cdf method



Simulating a loaded die using inverse cdf method

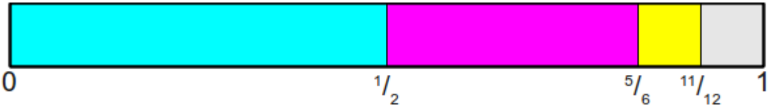


Pre-processing: compute $c_i = \sum_{j=0}^{i} p_j$ $\quad$ *c = cumsum(p)*

For each of the S samples:

1. Sample x uniformly from $[0, 1)$
2. Find minimum i such that $x < c_i$ $\quad$ $x \in [\sum_{j=0}^{i-1} p_j, \sum_{j=0}^{i} p_j)$

Simulating a loaded die using inverse cdf method



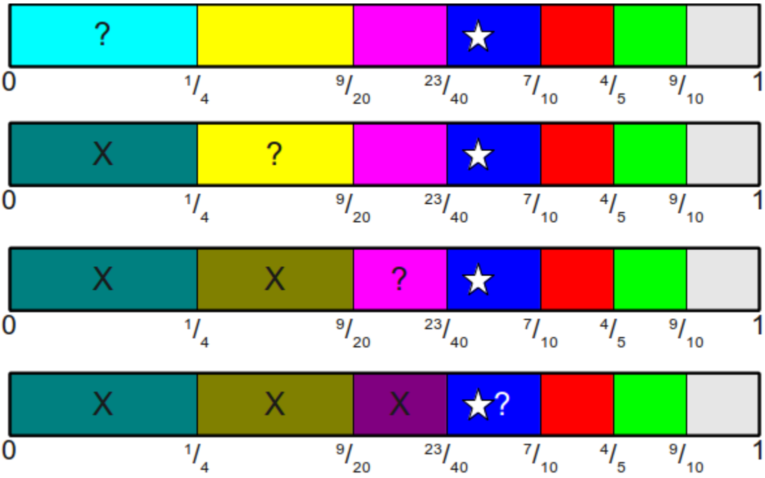
Pre-processing: compute $c_i = \sum_{j=0}^{i} p_j$ *c = cumsum(p)*

For each of the S samples:

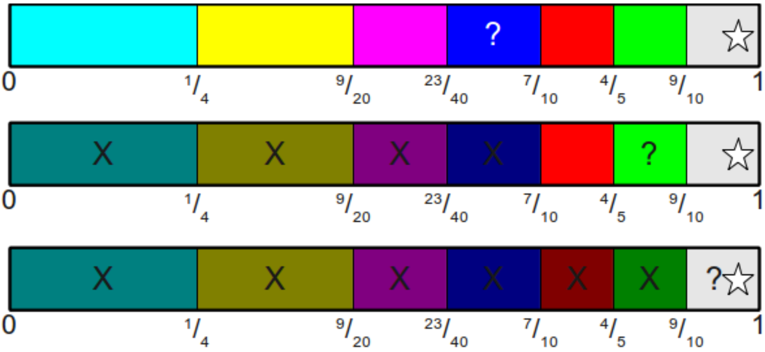
1. Sample x uniformly from $[0, 1)$
2. Find minimum i such that $x < c_i$ $x \in [\sum_{j=0}^{i-1} p_j, \sum_{j=0}^{i} p_j)$

- Generate uniform r. v. in $O(1)$
- Find bin not $O(1)$ anymore

Inverse CDF method: Linear search $O(n)$

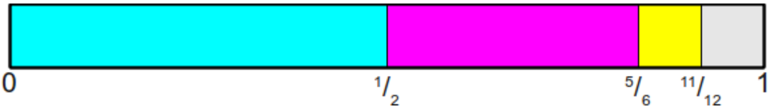


Inverse CDF method: Binary search $O(\log n)$

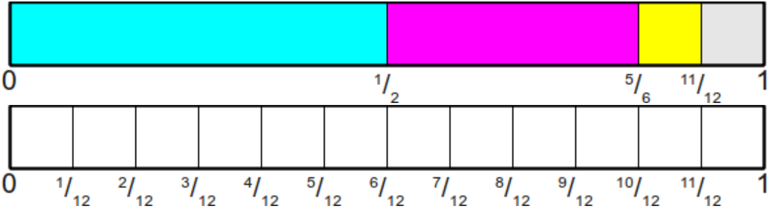



O(LOG N) IS NOT COOL, YOU KNOW WHAT'S COOL?
O(1)
memegenerator.net

Simulating a loaded die from a fair die - attempt 1

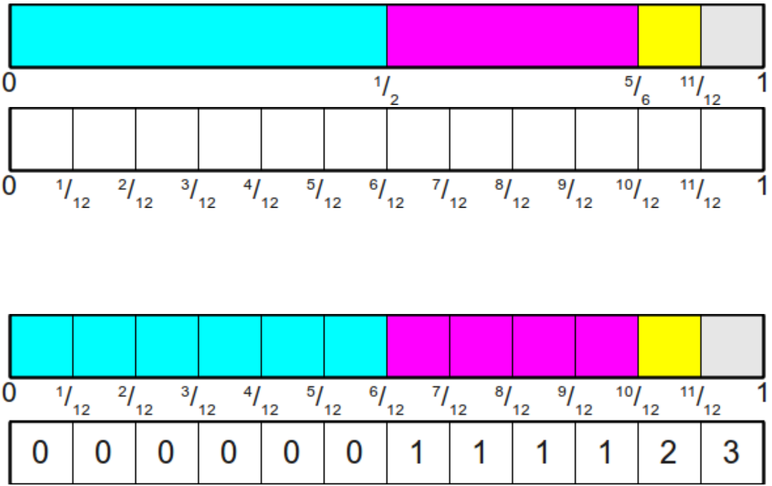


Simulating a loaded die from a fair die - attempt 1

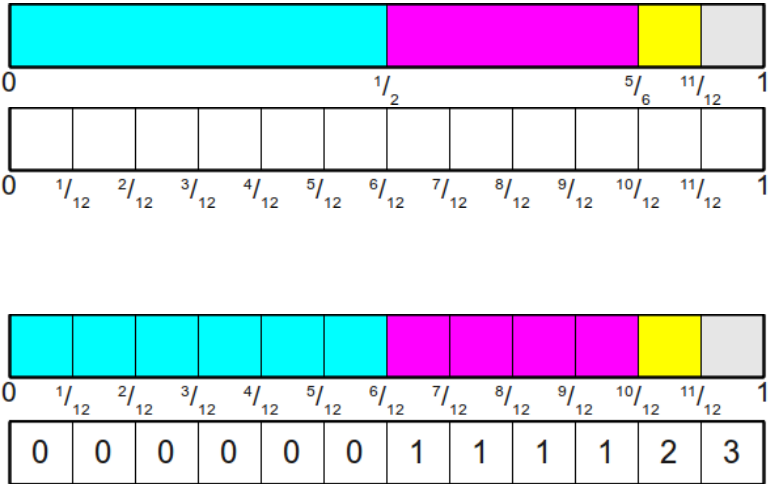


Simulating a loaded die from a fair die - attempt 1

Simulating a loaded die from a fair die - attempt 1



Simulating a loaded die from a fair die - attempt 1

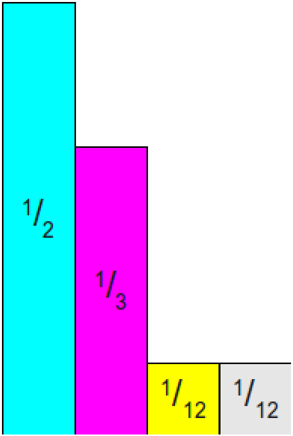


- Can sample in $O(1)$!
- Memory depends on LCM of the denominators: best $O(n)$, worst case $O(\prod_{i=1}^{n} d_i)$

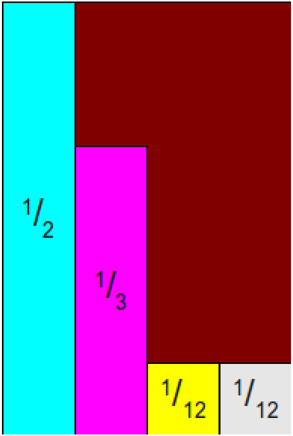
And now for something completely different ...

From 1 dimension to 2 dimensions

- $p = [1/2,\ 1/3,\ 1/12,\ 1/12]$
- say width = w and height p_i

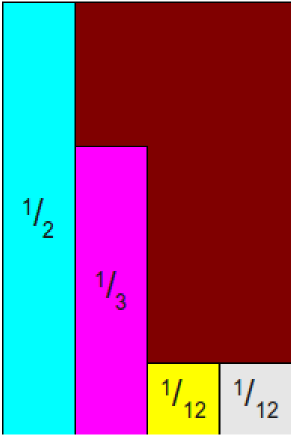


Darts



1. Throw dart
2. Return i if dart hits i^{th} rectangle, else go to step 1

Darts



$$Pr(\text{hit rectangle } i | \text{hit some rectangle}) = \frac{\text{area of rectangle } i}{\text{total area of valid rectangle}}$$
$$= \frac{wp_i}{w \sum_j p_j} = p_i$$

width w and height h don't matter

Say height is $h \cdot p_i$

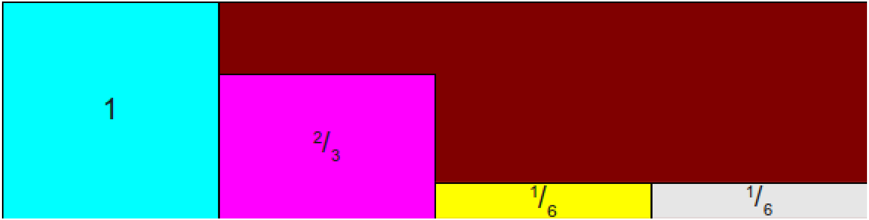
$$Pr(\text{hit rectangle } i | \text{hit some rectangle}) = \frac{\text{area of rectangle } i}{\text{total area of valid rectangle}}$$
$$= \frac{hwp_i}{hw} = p_i$$

width w and height h don't matter

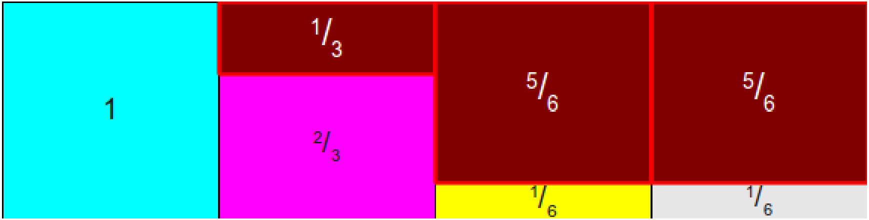
Say height is $h \cdot p_i$

$$Pr(\text{hit rectangle } i | \text{hit some rectangle}) = \frac{\text{area of rectangle } i}{\text{total area of valid rectangle}} = \frac{hwp_i}{hw} = p_i$$

Set $h = \frac{1}{p_{\max}}$ and $w = 1$ for convenience



A different way of simulating a loaded die



To generate a sample:

1. Choose i uniformly from one of n rectangles
2. Sample x uniformly from $[0, 1)$
3. If $x \leq \frac{p_i}{p_{\max}}$, return i, else go to step 1

Computational Complexity

- For each sample:
  1. Choose i uniformly from one of n rectangles
  2. Sample x uniformly from $[0, 1)$
  3. If $x \leq \frac{p_i}{p_{\max}}$, return i, else go to step 1

$Pr(\text{some side is chosen})$

$$= \sum_{i=0}^{n-1} \left(\frac{1}{n} \frac{p_i}{p_{max}}\right) = \frac{1}{n} \sum_{i=0}^{n-1} \frac{p_i}{p_{max}} = \frac{1}{n \cdot p_{max}} \sum_{i=0}^{n-1} p_i = \frac{1}{n \cdot p_{max}}$$

Computational Complexity

- For each sample:
  1. Choose i uniformly from one of n rectangles
  2. Sample x uniformly from $[0, 1)$
  3. If $x \leq \frac{p_i}{p_{\max}}$, return i, else go to step 1

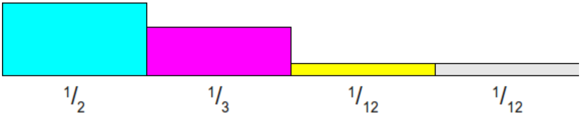
Pr(some side is chosen)

$$= \sum_{i=0}^{n-1} \left(\frac{1}{n} \frac{p_i}{p_{max}}\right) = \frac{1}{n} \sum_{i=0}^{n-1} \frac{p_i}{p_{max}} = \frac{1}{n \cdot p_{max}} \sum_{i=0}^{n-1} p_i = \frac{1}{n \cdot p_{max}}$$

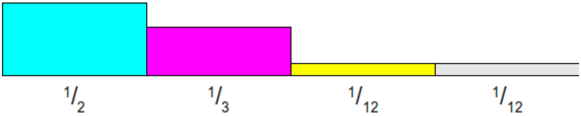
- Expected number of flips $= n \cdot p_{\max}$
- Best case: $O(1)$ for $p_{\max} = 1/n$
- Worst case: $O(n)$ for $p_{\max} = 1$

How do we improve on the worst case performance?

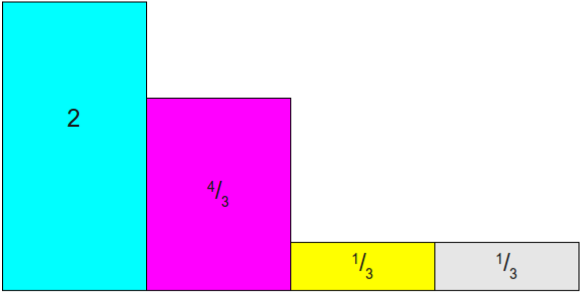
Alias method



Alias method

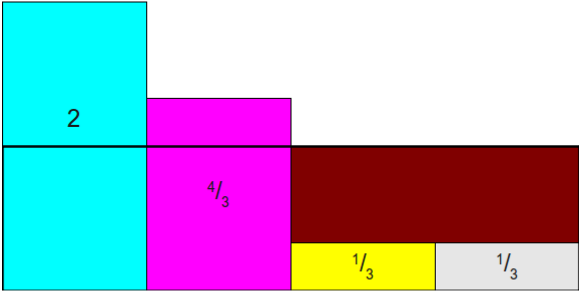


Set $h = \frac{1}{p_{ave}} = \frac{1}{4}$ instead of $h = \frac{1}{p_{\max}}$



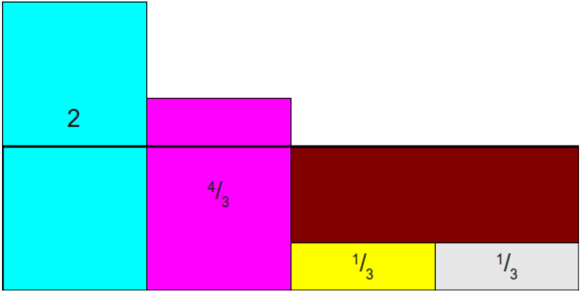
Alias method

Draw a horizontal line at height 1 and mark invalid regions in red



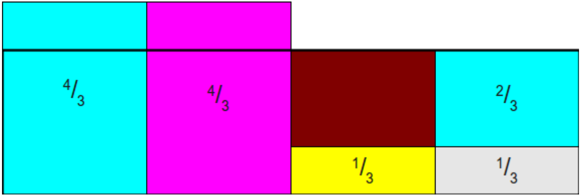
Alias method

Draw a horizontal line at height 1 and mark invalid regions in red

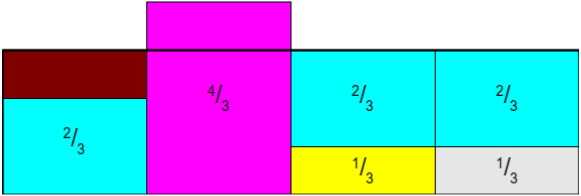


Key idea: Eliminate the wasteful red region such that each rectangle contains at most 2 valid colors.

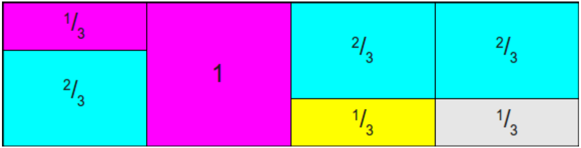
Alias method - 2



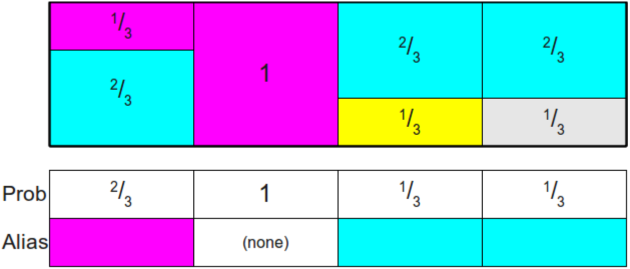
Alias method - 3



Alias method - 4



Alias method and corresponding alias table

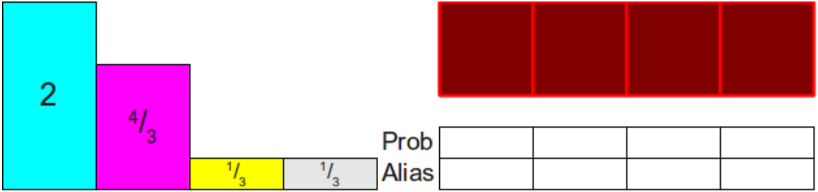


- Prob table contains height of *i* (probability of coin)
- Alias contains id of alternative color

Constructing alias tables

- An alias table can be constructed for any p
- $O(1)$ for sampling cost once alias table has been constructed
- Pre-processing cost for alias table construction:
  - Naive alias method: $O(n^2)$
  - Alias method: $O(n \log n)$
  - Vose's alias method: $O(n)$

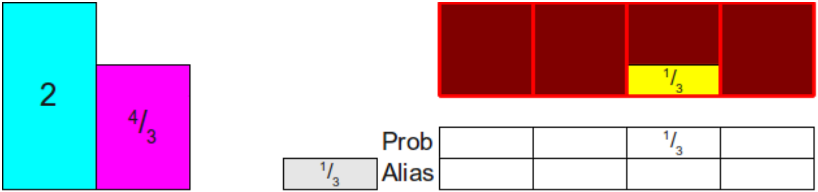
Naive alias table construction - 1



- Find some rectangle that has height at most 1 and place it into its own column, setting the Prob table to the height of that rectangle.
- Find some rectangle that has height at least 1 and use it to top off the column, setting the Alias table to correspond to the side of the die represented by the rectangle.

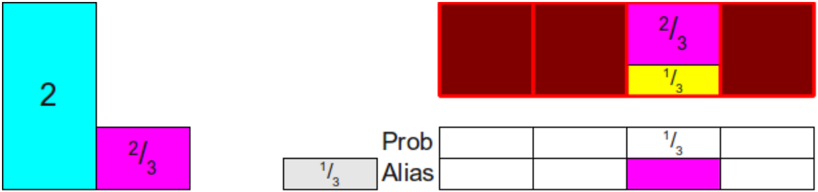
Naive alias table construction - 2

Fill Prob of column 3



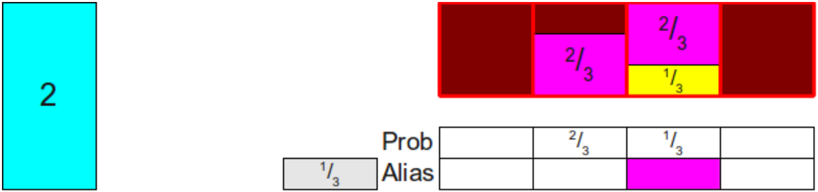
Naive alias table construction - 3

Choose column 2 as alias



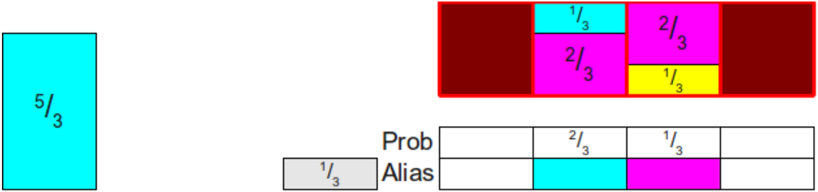
Naive alias table construction - 4

Fill Prob of column 2



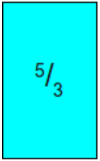
Naive alias table construction - 5

Choose column 1 as alias



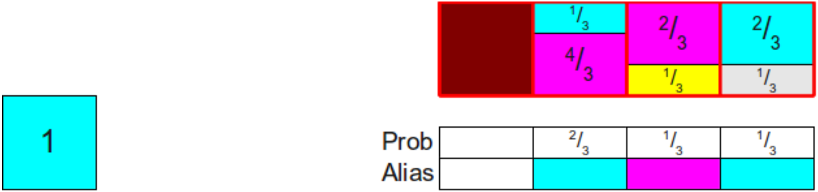
Naive alias table construction - 6

Fill Prob of column 4



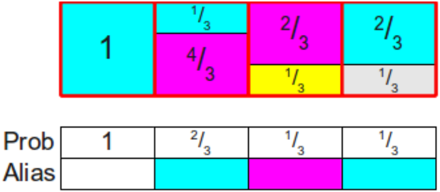
Naive alias table construction - 7

Choose column 1 as alias



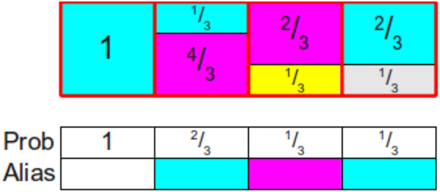
Naive alias table construction - 8

Fill Prob of column 1



Naive alias table construction - 8

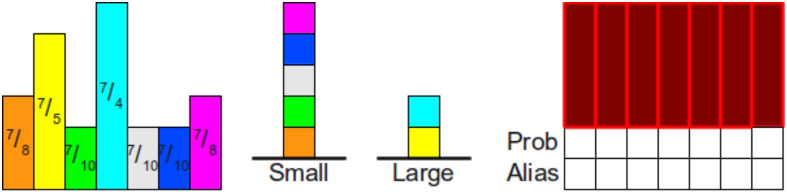
Fill Prob of column 1



- $O(n^2)$ using unsorted arrays
- $O(n \log n)$ using binary search tree
- $O(n)$ using Vose's method

Vose's method - 1

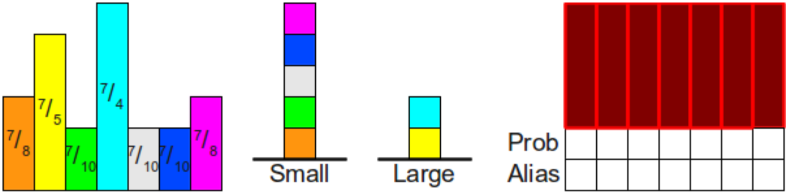
Consider $p = (\frac{1}{8}, \frac{1}{5}, \frac{1}{10}, \frac{1}{4}, \frac{1}{10}, \frac{1}{10}, \frac{1}{8})$



- Maintain two (unordered) stacks for small (height ≤ 1) and large (height > 1)

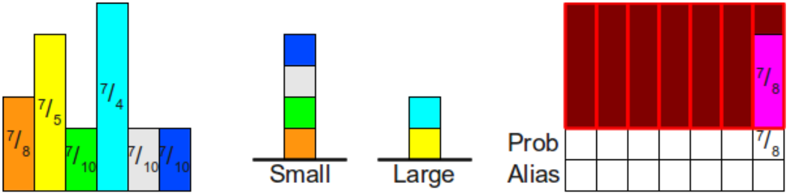
Vose's method - 1

Consider $p = (\frac{1}{8}, \frac{1}{5}, \frac{1}{10}, \frac{1}{4}, \frac{1}{10}, \frac{1}{10}, \frac{1}{8})$

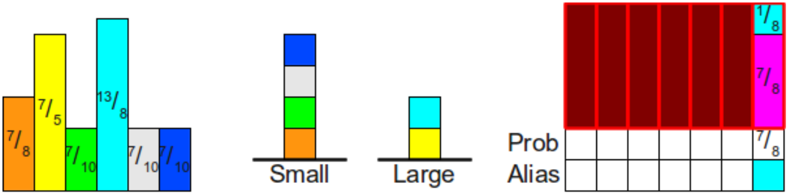


- Maintain two (unordered) stacks for small (height ≤ 1) and large (height > 1)
- Pseudocode:
    1. Pop top of small stack, say s, and fill in the corresponding prob column
    2. Pop top of large stack, say ℓ, to fill in the remaining $1 - s$
    3. If $\ell - (1 - s) \leq 1$, move $\ell - (1 - s)$ to top of the small stack
- Use of stack allows $O(n)$ construction

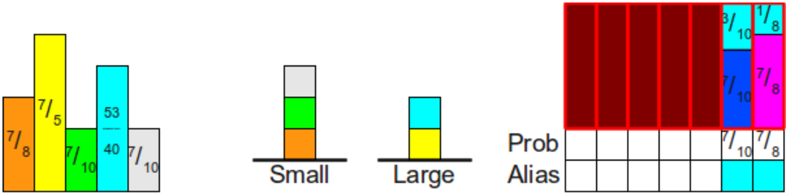
Vose's method - 2



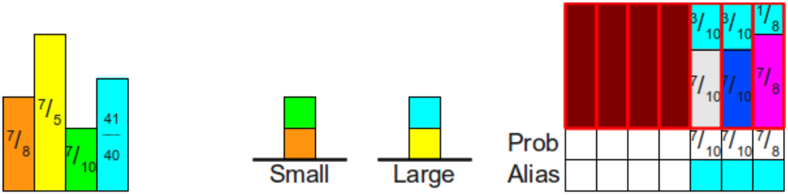
Vose's method - 3



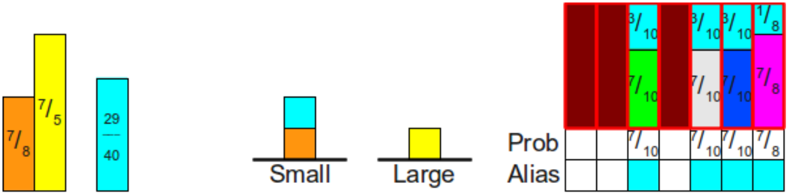
Vose's method - 4



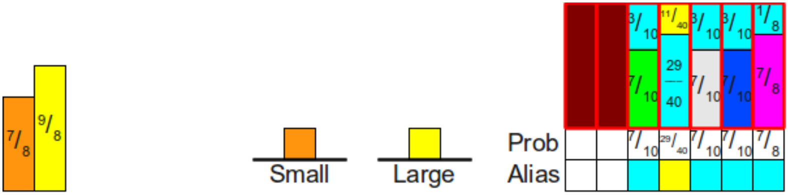
Vose's method - 5



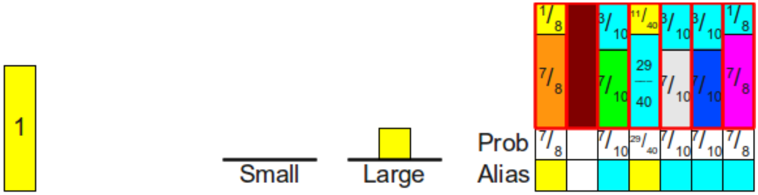
Vose's method - 6



Vose's method - 7



Vose's method - 8



Comparison of different methods

Algorithm	Initialization Time Best	Initialization Time Worst	Generation Time Best	Generation Time Worst	Memory Usage Best	Memory Usage Worst
Loaded Die from Fair Die	$\Theta(n)$	$O(\prod_{i=0}^{n} d_i)$	$\Theta(1)$		$\Theta(n)$	$O(\prod_{i=0}^{n} d_i)$
Loaded Die from Biased Coins	$\Theta(n)$		$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	
Roulette Wheel Selection	$\Theta(n)$		$\Theta(\log n)$		$\Theta(n)$	
Optimal Roulette Wheel Selection	$O(n^2)$		$\Theta(1)$	$O(\log n)$	$\Theta(n)$	
Fair Die/Biased Coin Loaded Die	$\Theta(n)$		$\Theta(1)$	$\Theta(n)$ (expected)	$\Theta(n)$	
Naive Alias Method	$O(n^2)$		$\Theta(1)$		$\Theta(n)$	
Alias Method	$O(n \log n)$		$\Theta(1)$		$\Theta(n)$	
Vose's Alias Method	$\Theta(n)$		$\Theta(1)$		$\Theta(n)$	

Thank you!