

The background features a complex network of thin grey lines connecting small black dots, forming a web-like structure on the left side. Scattered across the entire background are various sizes of triangles, some solid and some outlined, along with individual dots and small clusters of dots.

Reservoir Sampling

Sand Ip

382. Linked List Random Node

Medium



427



133



Favorite



Share

Given a singly linked list, return a random node's value from the linked list. Each node must have the **same probability** of being chosen.

Follow up:

What if the linked list is extremely large and its **length is unknown** to you?
Could you solve this efficiently without using extra space?



The background features a light gray gradient. In the top right corner, there is a complex network of thin black lines connecting various points, forming a web-like structure with several triangles of different sizes. Some points are solid black dots, while others are open circles. In the bottom left corner, there is a cluster of small, open circles of varying sizes, some of which are connected by faint lines.

Examine Google's stream of search queries and find a sample of what was asked over the last 24 hours

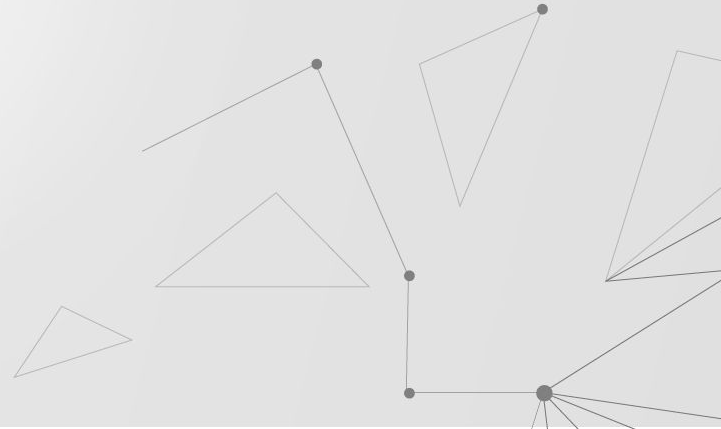
The background features a light gray gradient with abstract geometric elements. In the top right corner, there is a complex network of thin black lines connecting various points, some of which are small black dots. Several thin, light gray triangles are scattered throughout the upper right area. In the bottom left corner, there is a cluster of small, light gray circles and dots of varying sizes.

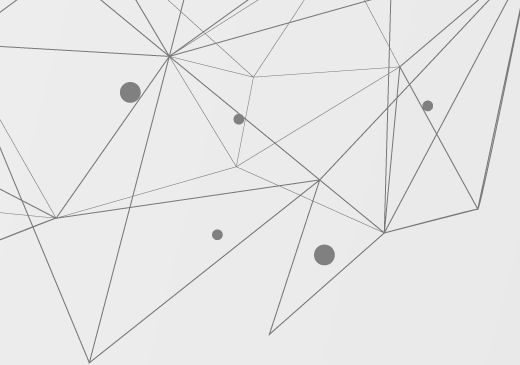
~63,000 queries per second
~5,443,200,000 queries per day



APPROACH

Implement an algorithm that randomly chooses 1000 items from this stream such that each item is equally likely to be selected





ANALYSIS

- - Data stream is large and will not fit into main memory
 - Can only iterate over data once
 - Unbiased selection of an item to be placed in sample

STEP 1

The first k events in the stream automatically enters the reservoir

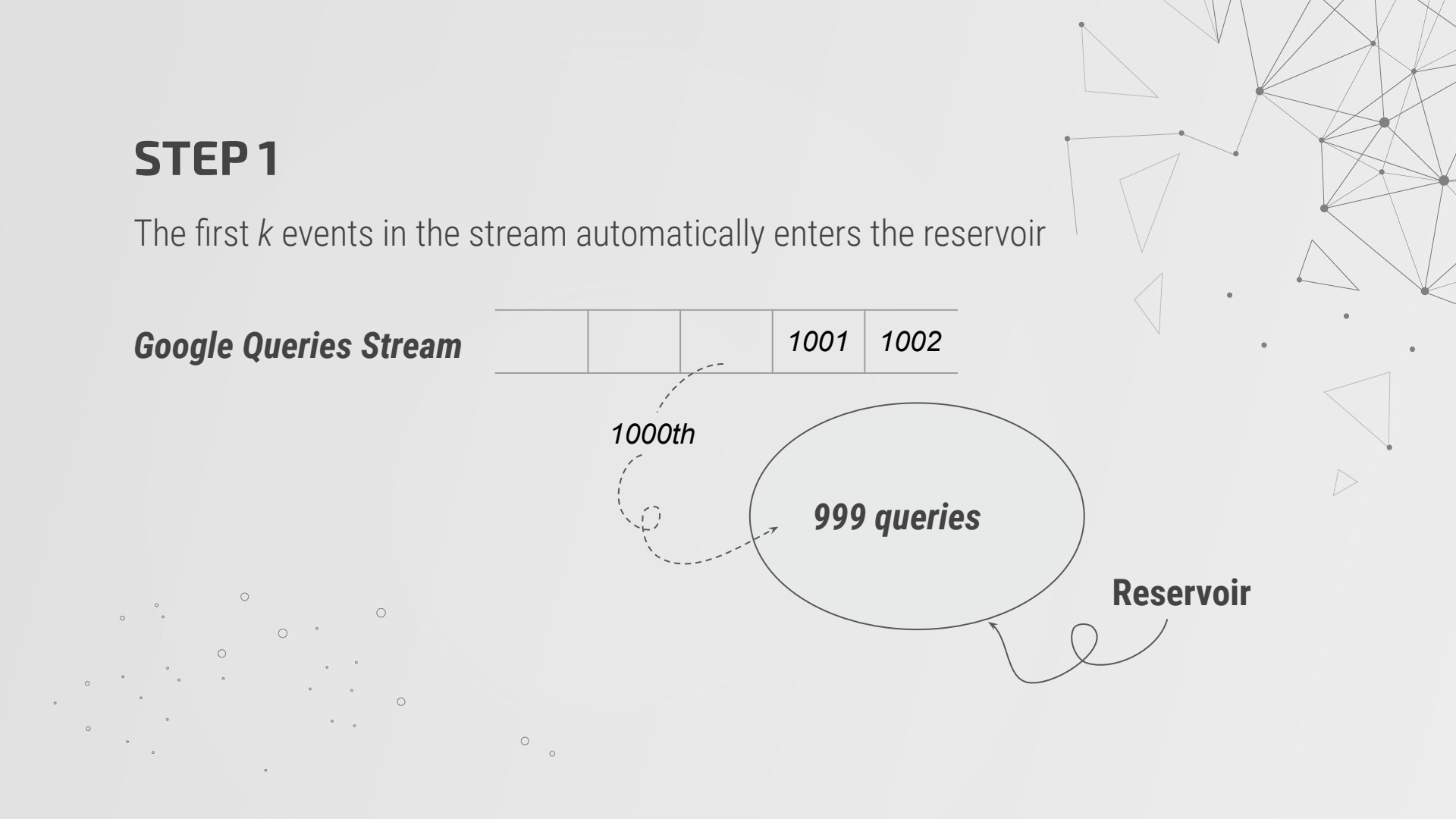
Google Queries Stream

			1001	1002
--	--	--	------	------

1000th

999 queries

Reservoir



STEP 2

For the i th event if $i > k$, draw a random number, q , between 0 and i .

If q is smaller than k (sample size=1000), randomly replace an element in the reservoir with the i th element

$i = 1001$

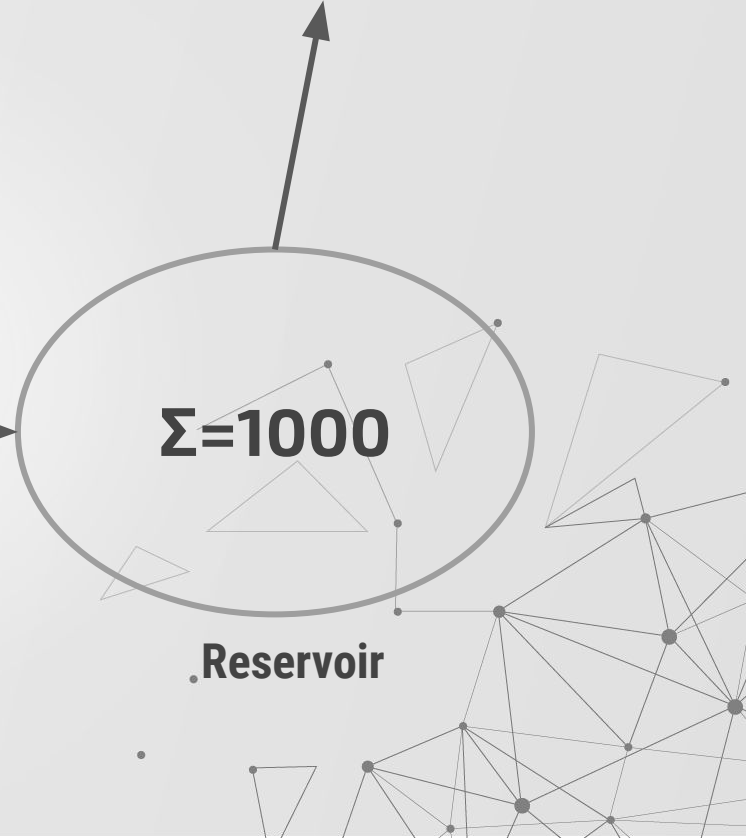
$\text{randint}(1, 1001) = 800$
 $800 < k$

$$\frac{k}{i} = \frac{1000}{1001} = 99.9\%$$

$\text{randint}(1, 1000) = 82$
Replace query 82 with query 1001

$\Sigma = 1000$

Reservoir



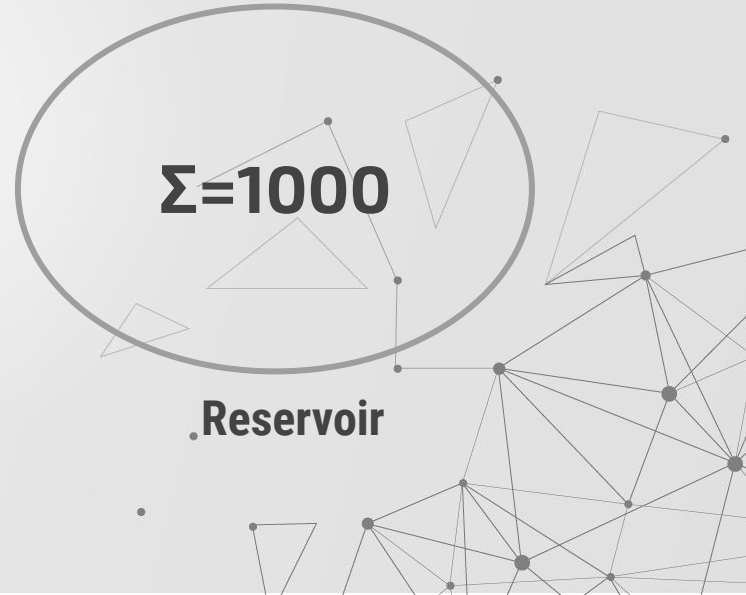
$i = 10,000$

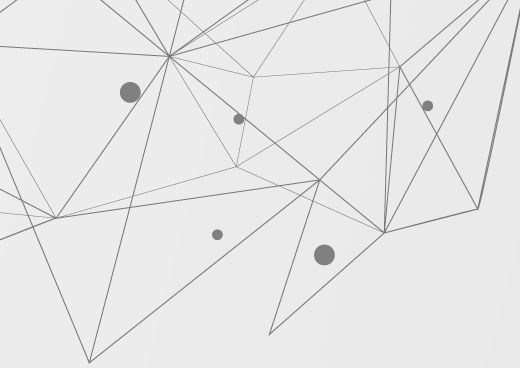
$\text{randint}(1, 10000) = 3100$
 $3100 \notin k$

$$\frac{k}{i} = \frac{\cancel{1000}}{\cancel{10000}} = 10\%$$

$\Sigma = 1000$

.Reservoir





- Over time, as i increases, the probability of it replacing an existing event in the reservoir decreases

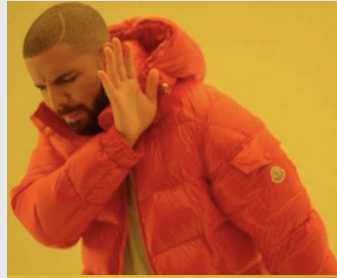
$O(n)$
 $O(1)$ per event



SOURCES

1. <https://pycon.org.il/2016/static/sessions/jonathan-arfa.pdf>
2. <https://www.kdnuggets.com/2019/09/5-sampling-algorithms.html>
3. <https://gregable.com/2007/10/reservoir-sampling.html>
4. <https://leetcode.com/problems/linked-list-random-node/>





10000



1001

