Department of Computer Science and Engineering (Data Science)

Machine Learning – IV

Experiment 4

Name: Umang Kirit Lodaya **SAP ID: 60009200032**

Batch: D11

Aim:

To implement and comprehend the Flajolet-Martin Algorithm, a probabilistic algorithm used for approximating the cardinality of a multiset or the number of distinct elements in a dataset.

Theory:

Introduction to Flajolet-Martin (FM) Algorithm:

- The Flajolet-Martin algorithm is a probabilistic method for estimating the cardinality of a set without requiring the set to be stored explicitly.
- It is particularly useful for large-scale data where maintaining an exact count of distinct elements is impractical.

Algorithm Overview:

- **Hash Functions:**
  - Utilize hash functions to map elements of the dataset into binary strings.
  - The hash functions should produce a large range of possible values.
- **Trailing Zeros:**
  - For each hashed value, determine the number of trailing zeros in its binary representation.
  - The maximum number of trailing zeros across all hash values provides an estimate of the logarithm base 2 of the cardinality.
- **Cardinality Estimation:**
  - Use the maximum number of trailing zeros to estimate the cardinality using the formula $2^{\max trailing\ zeros}$.

Step-by-Step Implementation:

- **Step 1: Hash Functions**
    - Implement hash functions that map elements to binary strings.
- **Step 2: Trailing Zeros Calculation**
    - For each hashed value, determine the number of trailing zeros in its binary representation.
- **Step 3: Cardinality Estimation**
    - Utilize the maximum number of trailing zeros to estimate the cardinality of the dataset.
- **Step 4: Experimentation with Hash Functions**
    - Explore the impact of different hash functions on the accuracy of the cardinality estimation.

Implementation Tips:

- Choose hash functions that distribute elements evenly to achieve better cardinality estimates.
- Experiment with multiple hash functions to improve accuracy.

Lab Experiments to be Performed in This Session:

Execute the FM algorithm on a dataset to gain insights into its functionality and operation.

NAME: UMANG KIRIT LODAYA

SAP ID: 60009200032

BATCH: K - K1 / D11

FLAJOLET MARTIN ALGORITHM

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
class FlajoletMartin:
    def __init__(self, x):
        self.x = x
        print(f"X: {x}")
        self._hash_function = input('ENTER HASH FUNCTION: ').strip()

    def __getR(self, b):
        if b == '0':
            return 0

        i = 0
        for n in b[::-1]:
            if n == '1':
                break
            i += 1

        return i

    def getDistinct(self):
        data = pd.DataFrame({'x': self.x})
        data['h(x)'] = data['x'].apply(lambda x: eval(self._hash_function, {
        data['Binary'] = data['h(x)'].apply(lambda x: bin(x)[2:] if x != 0 e
        data['r'] = data['Binary'].apply(lambda x: self.__getR(x))

        print()
        display(data)
        print()

        R = max(data['r'])
        print(f"\nValue of R = {R}")
        print(f"Number of Distinct Elements = {2 ** R}")
```

In [3]:
```
x = [1, 3, 2, 1, 2, 3, 4, 3, 1, 2, 3, 1]
# H = ((6 * x) + 1) % 5

FM = FlajoletMartin(x)
FM.getDistinct()
```

```
X: [1, 3, 2, 1, 2, 3, 4, 3, 1, 2, 3, 1]
ENTER HASH FUNCTION: ((6 * x) + 1) % 5
```

	x	h(x)	Binary	r
0	1	2	10	1
1	3	4	100	2
2	2	3	11	0
3	1	2	10	1
4	2	3	11	0
5	3	4	100	2
6	4	0	0	0
7	3	4	100	2
8	1	2	10	1
9	2	3	11	0
10	3	4	100	2
11	1	2	10	1

```
Value of R = 2
Number of Distinct Elements = 4
```

In [4]:

```
x = [1, 5, 10, 5, 15, 1]
# H = x % 11

FM = FlajoletMartin(x)
FM.getDistinct()
```

```
X: [1, 5, 10, 5, 15, 1]
ENTER HASH FUNCTION: x % 11
```

	x	h(x)	Binary	r
0	1	1	1	0
1	5	5	101	0
2	10	10	1010	1
3	5	5	101	0
4	15	4	100	2
5	1	1	1	0

```
Value of R = 2
Number of Distinct Elements = 4
```