

**Terna Engineering College**  
**Computer Engineering Department**

Program: Sem VII

**Course: Big Data Analytics & Computational Lab -I (BDA&CL-I)**

**Experiment No. 08**

**PART B**

**(PART B: TO BE COMPLETED BY STUDENTS)**

*(Students must submit the soft copy as per the following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Blackboard access available)*

Roll No. 50	Name: AMEY THAKUR
Class: BE-COMPS-50	Batch: B3
Date of Experiment: 05-10-2021	Date of Submission: 05-10-2021
Grade :	

**Aim:** To implement the DGIM algorithm.

**B.1 Software code written by a student:**

```
import os
import time

def add_1(data, idx, max_key, cnt):
    data[0] = [[idx, cnt]] + data[0]
    current = 0
    while len(data[current]) > 2:
        assert len(data[current]) == 3
        buckets = data[current][-2:]
        data[current] = data[current][:1]
        current += 1

    if data.get(current, None) is None:
        data[current] = [buckets[0]]
        max_key = current
        # assert max_key == current
    else:
        data[current] = [buckets[0]] + data[current]
    return max_key
```

```

def output_data(data, max_key):
    cnt = 0
    mul = 1
    for i in range(max_key + 1):
        # print(len(data[i]) * mul)
        cnt += len(data[i]) * mul
        mul *= 2
    print("DGIM Count:", cnt - mul // 4)
    return cnt - mul // 4

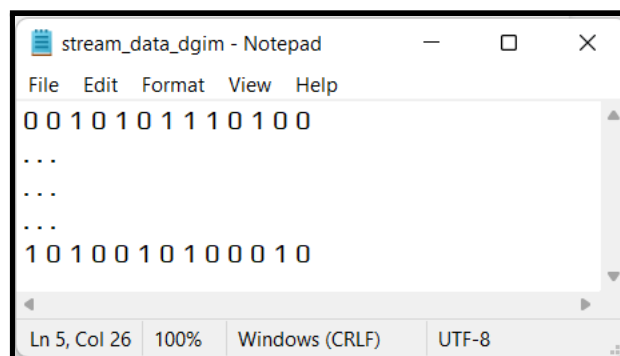
cnt = 0
idx = 0
bucket_dict = {0: []}
max_key = 0
window_size = 1000

start = time.time()
with open("/content/stream_data_dgim.txt", "r") as f:
    while True:
        input_bit = f.read(1)
        if not input_bit:
            DGIM_count = output_data(bucket_dict, max_key)
            break
        if input_bit == "\t":
            continue
        if len(bucket_dict[0]) > 0 and idx == bucket_dict[max_key][-1][0]:
            bucket_dict[max_key] = bucket_dict[max_key][:-1]
            if len(bucket_dict[max_key]) == 0:
                bucket_dict.pop(max_key)
                max_key -= 1
        if input_bit == '1':
            max_key = add_1(bucket_dict, idx, max_key, cnt)
            cnt += 1
        idx = cnt % window_size

end = time.time()
print("DGIM Run Time(s): %.4f" % (end - start))

```

## B.2 Input and Output:



```
DGIM Count: 508
DGIM Run Time(s): 0.0890
```

## B.3 Observation and learning:

DGIM is an efficient algorithm in processing large streams. When it's infeasible to store the flowing binary stream, DGIM can estimate the number of 1-bits in the window.

## B.4 Conclusion:

Successfully implemented DGIM algorithm.

## B.5 Questions of Curiosity:

1. Employ the DGIM algorithm. Shown below is a data stream with  $N=24$  and the current bucket config. A new element enters the window at the right. Thus, the oldest bit of the window is the left-most bit shown.

101011000101110110010110

- A. What is the largest bucket size for  $N=24$ ?
- B. Show one way of how the above initial stream will be divided into buckets.
- C. What is the estimate of the no. of 1's in the latest  $k=14$  bits of this window?

**Ans:**

- A. 4
- B.

$$\frac{101011}{2^2 = 4} \quad 000 \quad \frac{10111}{2^2} \quad 0 \quad \frac{11}{2^1} \quad 00 \quad \frac{101}{2^1} \quad \frac{1}{2^0} \quad 0$$

- C. 8