Experiment 3

DATE:

AA Experiment 3

Kreena Shah
60004210243
C'32

Aim : To perform amortized analysis using potential method for incrementing binary counter

Theory :

Potential method of amortized analysis is associated with the data structure as a whole rather than with specific objects within the data structure

The amortized cost $\hat{c}_i$ of the i^{th} operation with respect to potential function ϕ is defined by :

$$\hat{c}_i = \underset{\text{actual}}{c_i} + \underbrace{\phi(D_i) - \phi(D_{i-1})}_{\text{potential change}}$$

Algorithm

i ← 0
while (i < length(A) and A(i) == 1)
  do A(i) ← 0
    i ← i+1
if i < length(A)
then A(i) ← 1

Value Counter	A[2]	A[1]	A[0]	Actual cost	$\phi(D_i)$	$\phi(D_{i-1})$	$\Delta\phi$	Amortized cost
0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	0	1	2
2	0	1	0	2	1	1	0	2
3	0	1	1	1	2	1	1	2

DATE:

Conclusion : Thus, we implemented incrementing binary counter using potential method of amortized analysis

FOR EDUCATIONAL USE

Code :

```
binaryStr = "0000"
binaryPrev = binaryStr

# cost to flip 0 -> 1 always = 1
```

```
setCost = 1
# cost to flip required 1s -> 0s
resetCost = 0
# total cost -> setCost + resetCost
totalCost = 0

potential = 0
potentialPrev = 0
amortized = 0

print("Element Set   Reset   Total  Potential Amortized")

for i in range(8):
    if i == 0:
        setCost = 0
    else:
        setCost = 1
        resetCost = 0
        binaryPrev = binaryStr
        binaryStr = bin(i)[2:].zfill(4)
        for j in range(len(binaryStr)):
            if binaryPrev[j] == '1' and binaryStr[j] == '0':
                resetCost += 1
        totalCost = resetCost + setCost
        potentialPrev = potential
        potential = potentialPrev - resetCost + setCost

    amortized = totalCost + potential - potentialPrev
    print(f"{binaryStr}\t {setCost}\t\t{resetCost}\t\t{totalCost}\t\t{potential}\t\t{amortized}")
```

Element	Set	Reset	Total	Potential	Amortized
0000	0	0	0	0	0
0001	1	0	1	1	2
0010	1	1	2	1	2
0011	1	0	1	2	2
0100	1	2	3	1	2
0101	1	0	1	2	2
0110	1	1	2	2	2
0111	1	0	1	3	2