P1: B+ Tree

1.1 Block size = 2 \* d \* key\_size + (2 \* d + 1) \* pointer\_size

= 2 \* 5 \* 8 + (2 \* 5 + 1) \* 16

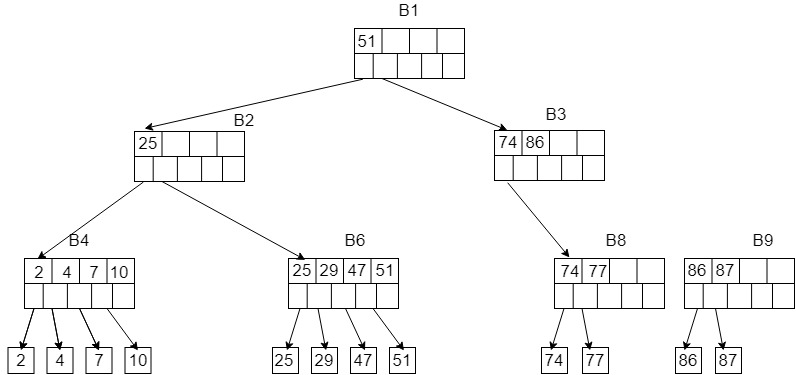
= 256 bytes

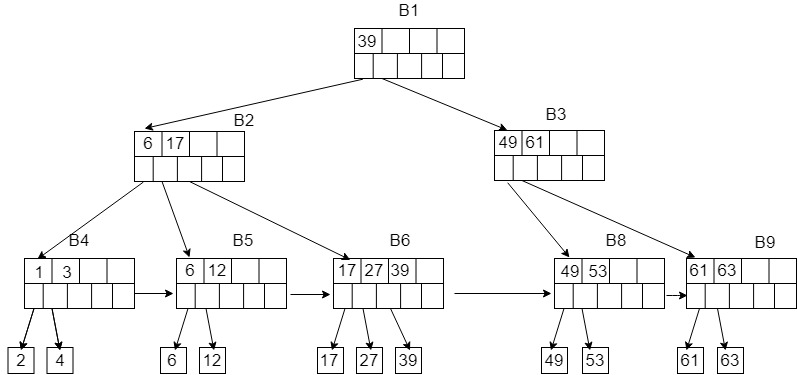
2.1 key 29: B1, B2, B6

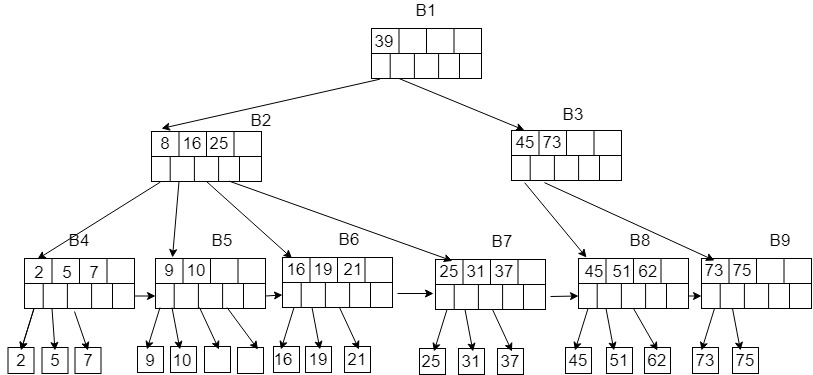
key 61: B1, B3, B7, B8

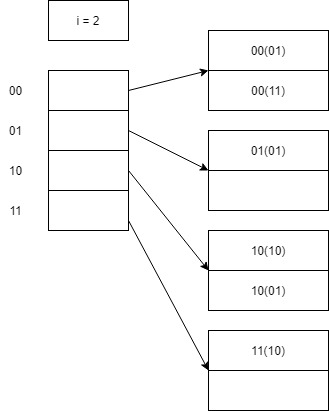
2.2 B1, B2, B4, B5, B6, B7, B8

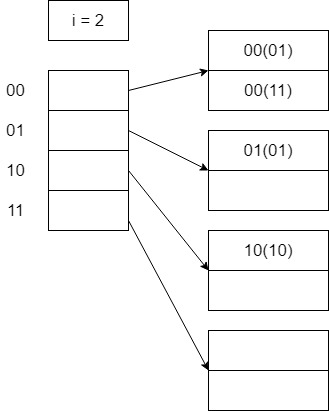
3.1



3.2

3.3

P2. a b



1. After inserting 0001: b. After inserting 1110:

i = 2, A blank bucket is filled with it.

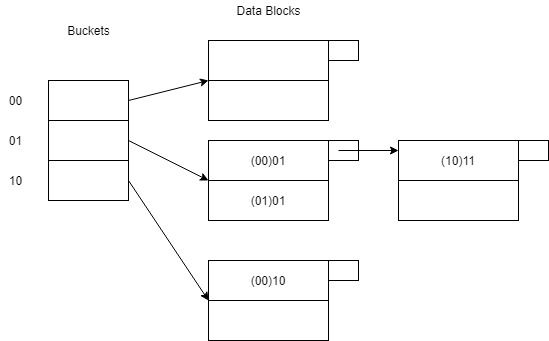
Split the above block,

Two buckets are added. After inserting 1001:

A blank bucket is filled with it.

After insering 0011:

A blank block is filled with it.

P3. After inserting 0001:

1. N = 3, r = 4,

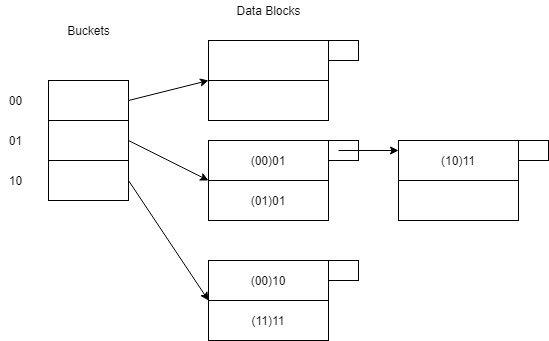
A new bucket is added,

Old μ = r/(N-2) = 4 /(3 \* 2 -2) = 1 > 0.9

New μ = r/N = 4/(3 \* 2) = 0.67 < 0.9

(2) Cause bit flips in this insertion

(3) i = 2

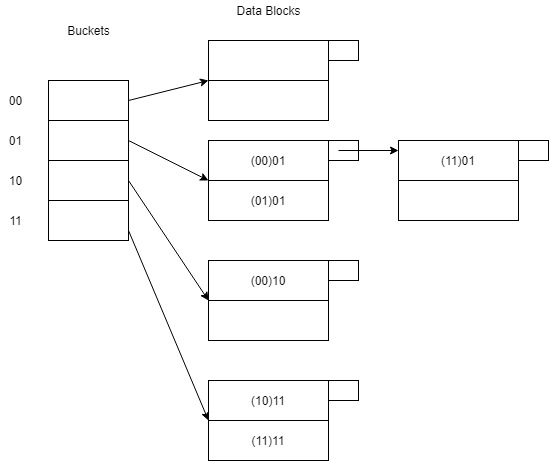
After inserting 1111:

(1)N = 3, r = 5

μ = 5 / (3 \* 2) = 0.83 < 0.9

(2) Cause a bit filp in this insertion

(3) i = 2

After inserting 1101:

(1)N = 4, r = 6

A new bucket is added

Old μ = 6 / （4 \* 2 – 2） = 1 > 0.9

New μ = 6 / （4 \* 2） = 0.75 < 0.9

(2) No bit flip.

(3) i = 2