Hashing: (bucket size = 2)

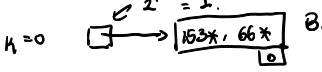
start with empty bucket., k=0

∵ k=0 ∴ no bits are used, all entries are in one bucket.

1. Insert 153* (key) 10011001 ← h(key)

66* 01000010.

$2^{k=0} = 1$.



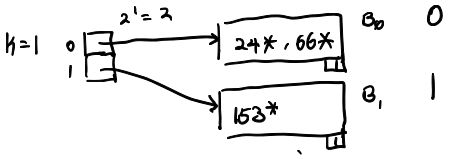
2. Insere 24* 00011000 → no room, k = B_0.level=0.

Double the directory.

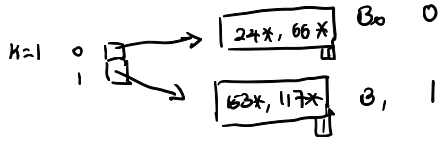
++ B_0.level.

split B_0 using last k=1 bits,

this is determined by the inserted key, h_k(key).



3. Insert 117* 01110101



4. Insert 4*, 00000100, last k=1 bits → 0.

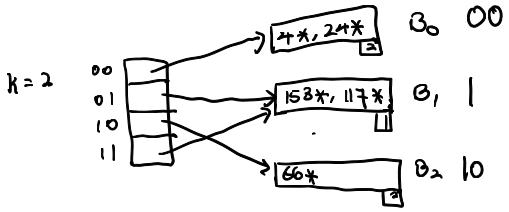
Insert to B_0 but no room, k = B_0.level = 1

Double the directory,

++ B_0.level,

Split B_0 using the last k=2 bits,

* Only touch the bucket that is inserted



01 and 11 are both pointed B_1 because B_1 only use last 1 bit. B_1.level = 1.

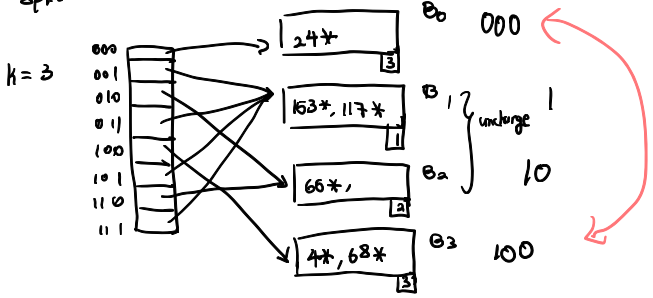
5. Insert 68* 01000100 → B_0 is full

↳ k = B_0.level = 2.

∴ Double the directory ++ k

++ B_0.level

split B_0 using the last B_0.level = 3 bits.



000 points to B_0.

001, 011, 101, 111 point to B_1 → B_1 remains the same

↳ B_1 still uses the last 1 bit, 1

010, 110 point to B_2

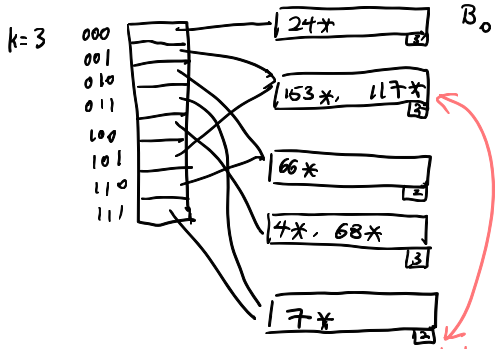
↳ B_2 remains the same

↳ B_2 still uses the last 2 bits, 10

One thing to notice that is that some address spaces grow but some remains the same.

Inserted 7* 00000111

B_1 is full → k=3 > B_1.level = 1, no need to enlarge directory,



++ B_2.level → each time splitting the bucket, bucket l.v. use another bucket increase by 2.

whenever use a new bucket, that buckets level = the bucket level of bucket being split after increasing the bucket level.