DCS Assignment 1

It is given that most of the user queries involve the Author_Name and/or Book Name.
So, we can use first 3 letters of the Book Name to design hash function.

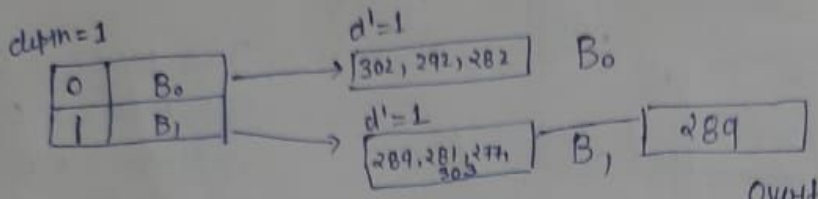
Hash function used Sum of ASCII value of first three characters.

Book First 3 characters	Asscii Value (sum)	Binary
How	302	100101110
Sel	292	100010000
Mal	282	100010010
Emo	289	100100001
Pha	281	100011001
Fan	277	100010101
Gui	303	100101111
The	289	100100001
Wod	298	100101010

for finding Sum of Asscii value we consider first letter capital/~~small~~ uppercase rest two in lower case.

(a) Bucket size : 4

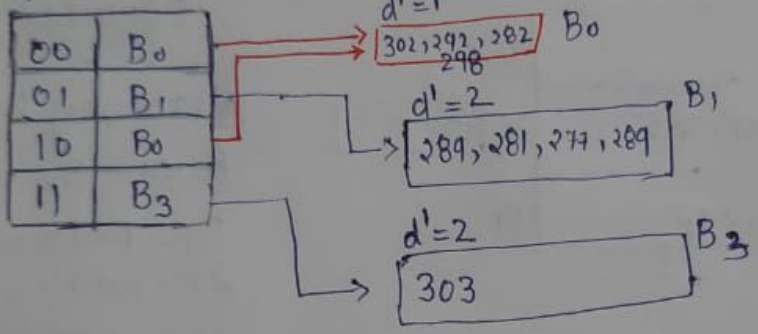
Step 1 : mod. function used : key mod 2



value inserted	
302 mod 2 = 0	B_0
292 mod 2 = 0	B_0
282 mod 2 = 0	B_0
289 mod 2 = 1	B_1
281 mod 2 = 1	B_1
277 mod 2 = 1	B_1
303 mod 2 = 1	B_1
289 mod 2 = 1	B_1

(Overflow)
Now How to split

Step 2 depth = 2 hash function : (key) mod 4

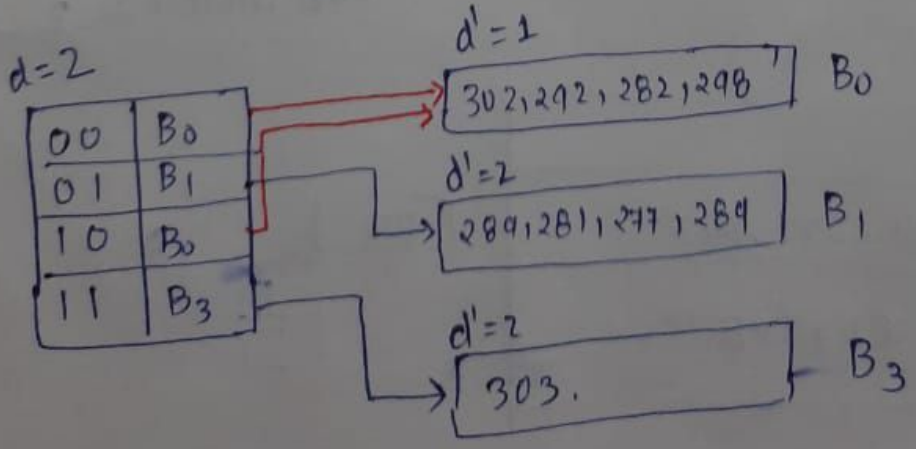


value inserted after splitting.

289 mod 4 = 1 = B_1
281 mod 4 = 1 = B_1
277 mod 4 = 1 = B_1
298 303 mod 4 = 3 = B_3
289 mod 4 = 1 = B_1

Now New value
298 mod 4 = 2 = B_0
since B_2 is pointing to B_0

Final structure.

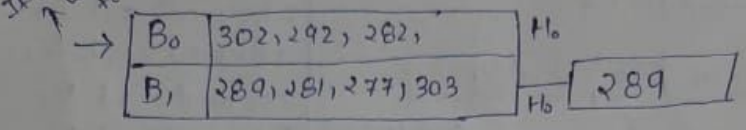


(b) To be inserted 302, 292, 282, 289, 281, 277, 303, 289, 298 ③

(1) Number of bucket = 2 Bucket size = 4

H_0 = key mod 2

It denotes which bucket to be split

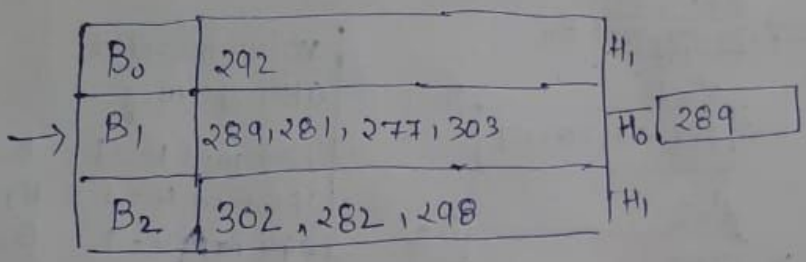


Overflow

302 mod 2 = 0 B_0
292 mod 2 = 0 B_0
282 mod 2 = 0 B_0
289 mod 2 = 1 B_1
281 mod 2 = 1 B_1
277 mod 2 = 1 B_1
303 mod 2 = 1 B_1
289 mod 2 = 1 B_1
(Overflow)

So split B_0 to B_0 and B_3

New Hash function H_1 = key mod 4



Arranging keys of B_0 and B_2 using H_1

302 mod 4 = 2 B_2
292 mod 4 = 0 B_0
282 mod 4 = 2 B_2

Now insert 298

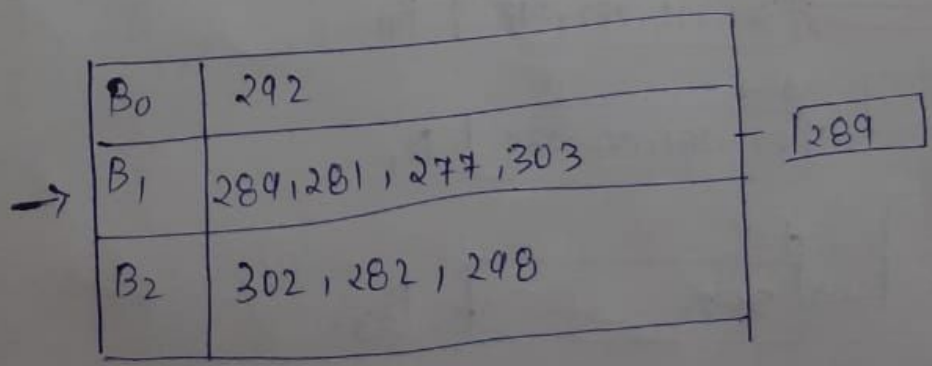
298 mod 2 = 0

∴ It is directed B_0 which is using H_1

So again to check between B_0 and B_2

298 mod 4 = 2 = B_2

Final

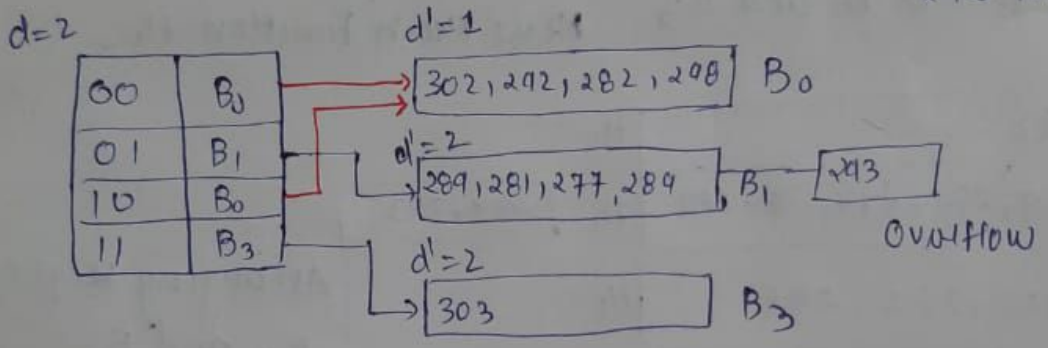


(c) record < MMi-009; Mi009_Soc, Marvin Minsky, Society of Mind >

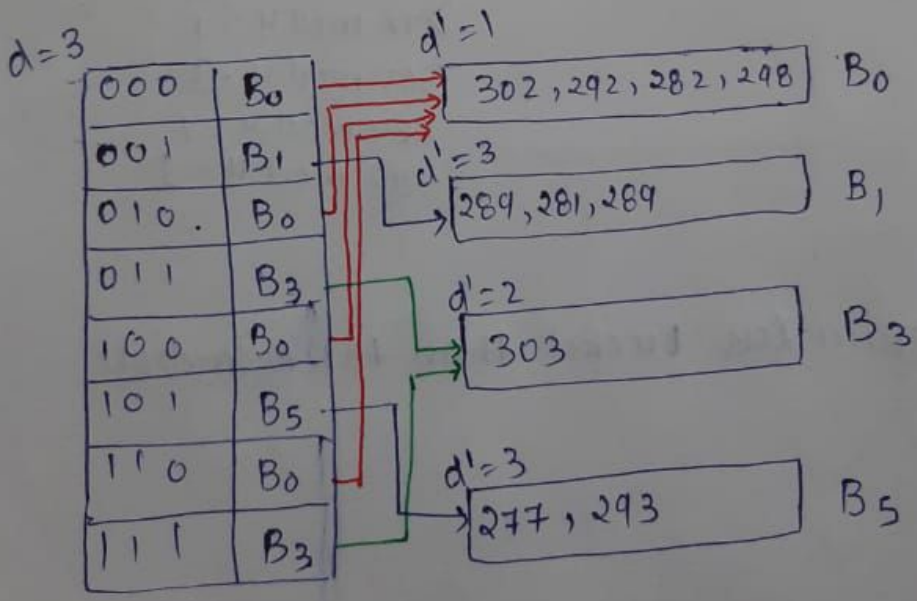
Book first 3 char	Ascii value	Binary.
Soc	83+111+99 = 293	100100101

Adding into structure of Question (a)

293 mod 4 = 1 B_1



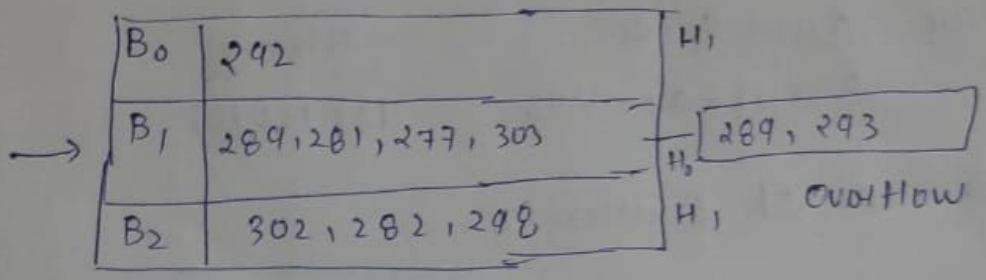
Hash function : (key) mod 8



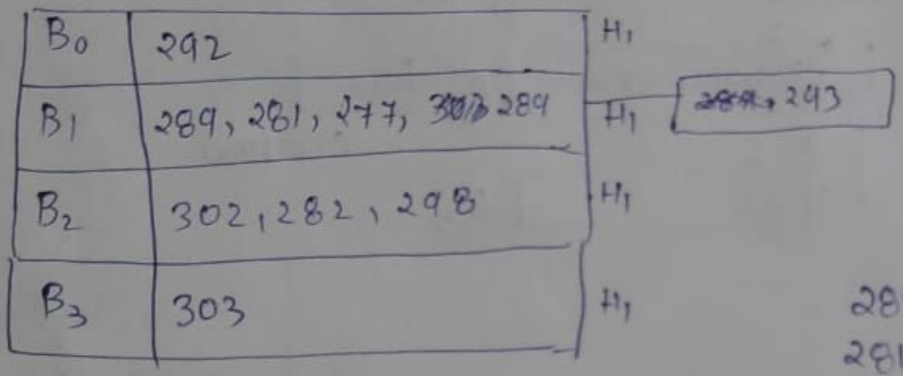
289 mod 8 = 1 B_1
281 mod 8 = 1 B_1
277 mod 8 = 5 B_5
289 mod 8 = 1 B_1
293 mod 8 = 5 B_5

Redistribution b/w B_1 and B_3

Adding 293 in structure of Question 6.



So split B_1 to B_1 and B_3 New Hash function H_1



Arranging keys of B_1 and B_3

289 mod 4 = 1 B_1
281 mod 4 = 1
277 mod 4 = 1
303 mod 4 = 3
289 mod 4 = 1
293 mod 4 = 1

Comparison:

* Linear hashing requires less bucket than the extendible hashing. only bucket needed
* No directory needed in linear hashing, whereas both directory and bucket needed in extendible hashing.