UMass Lowell
Department of Computer Science
Spring 2017

Instructor: Dr. Cindy Chen
TA: Xinzi Sun

COMP.5740 MIDTERM
March 7, 2017

2.5 hours
Closed book, closed notes

Name: Duy Truong

Student ID: 01489661

Problem	**Score**	
1	(17%)	9
2	(21%)	15
3	(62%)	36
Total	(100%)	60

NOTE: Please write clearly.

Problem 1

(17 points)

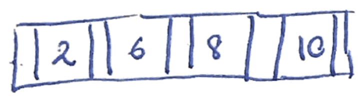
A (10 points)

Construct a B+-tree for the following search-key values:

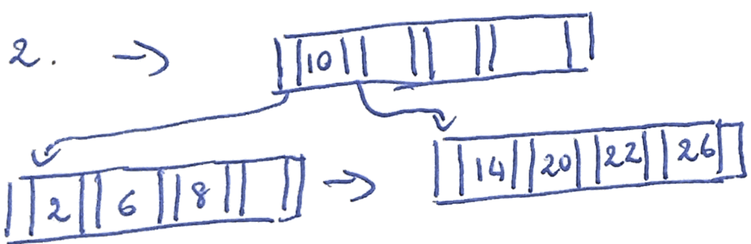
`2, 6, 8, 10, 14, 20, 22, 26, 32, 34`

Assume that the tree is initially empty and values are added in the order shown. Also assume that the number of pointers that will fit in one node is **four**. Show step by step results and the final B+-tree constructed.

step 1. →

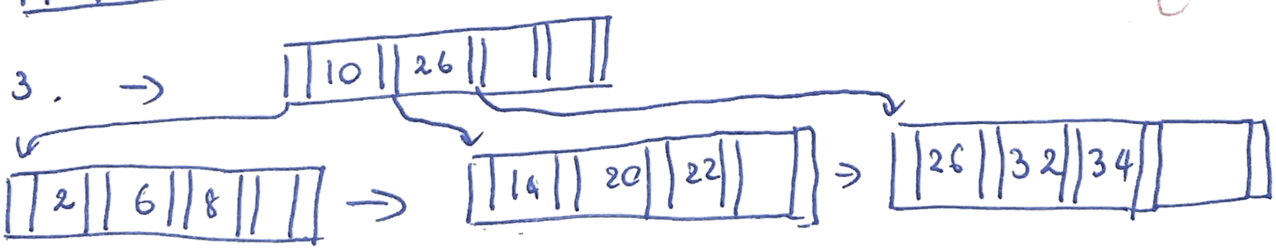


2. →



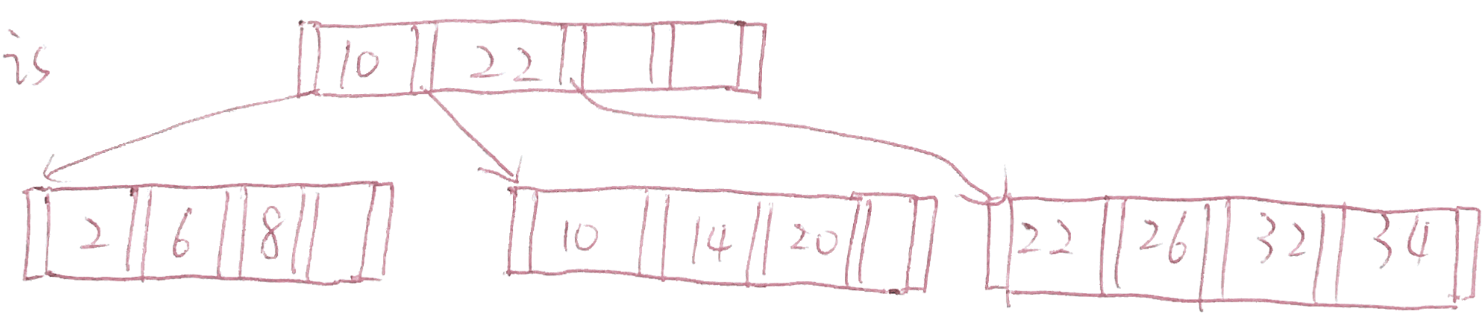
−8

3. →



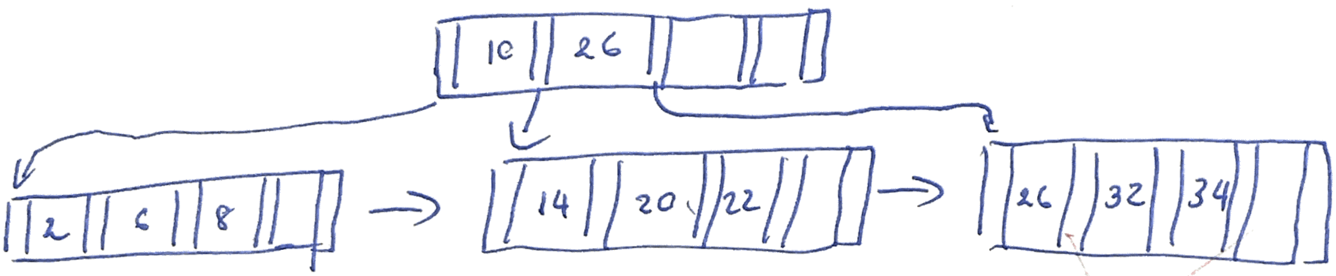
For five pointers the right answer

is

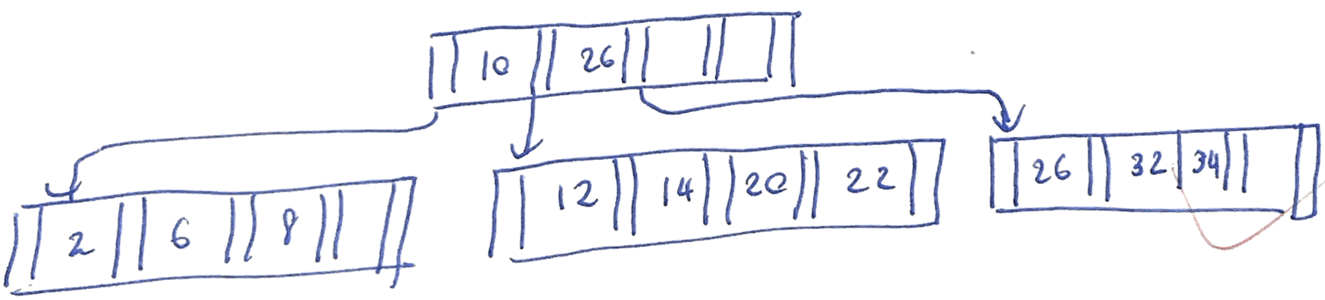


B (3 points) Insert into the B+-tree just constructed in Part [A] three new records with the following key values: 12, 13, 11. (The records are inserted in the order shown.) Draw the resulting B+-tree after each insertion.

Part [A]

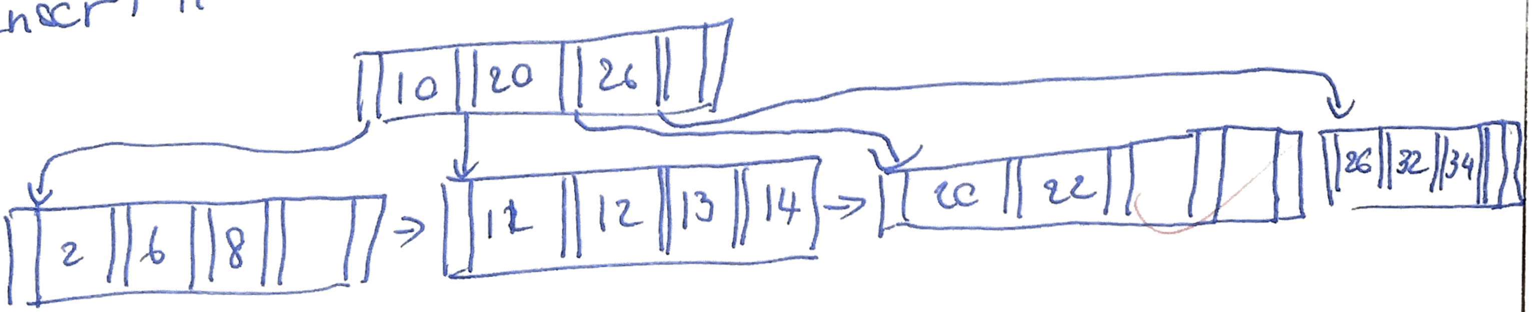


Insert 12:



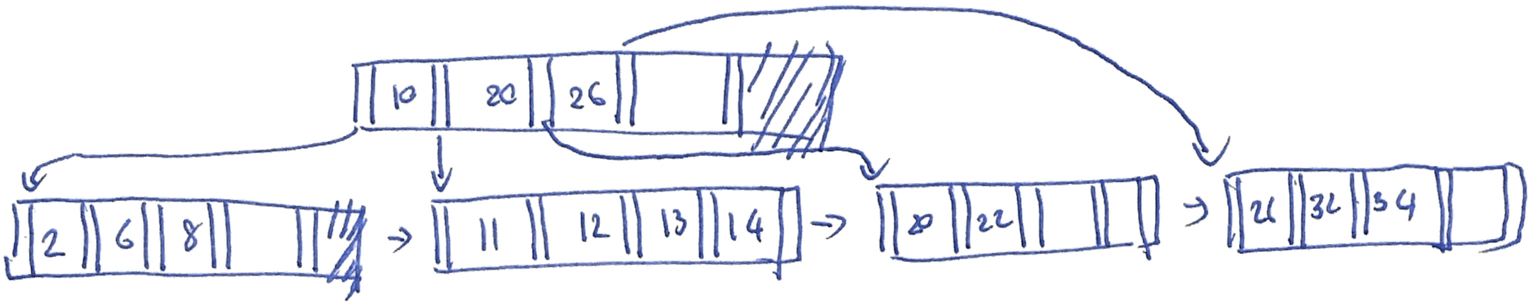
Insert 13

Insert 11

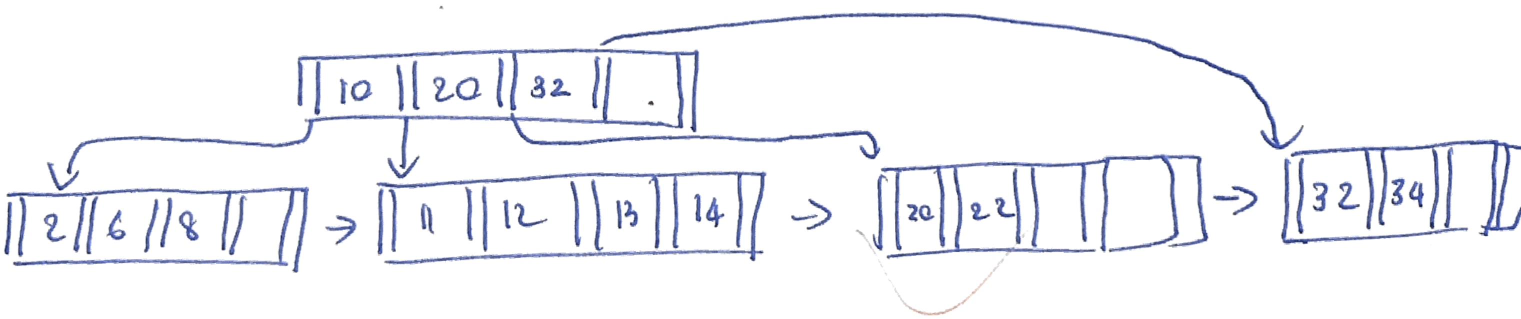


C (4 points) Now, from the B+-tree obtained in Part [B], first delete the record with key value 26 and then the record with key value 22. Draw the resulting B+-tree after each deletion.

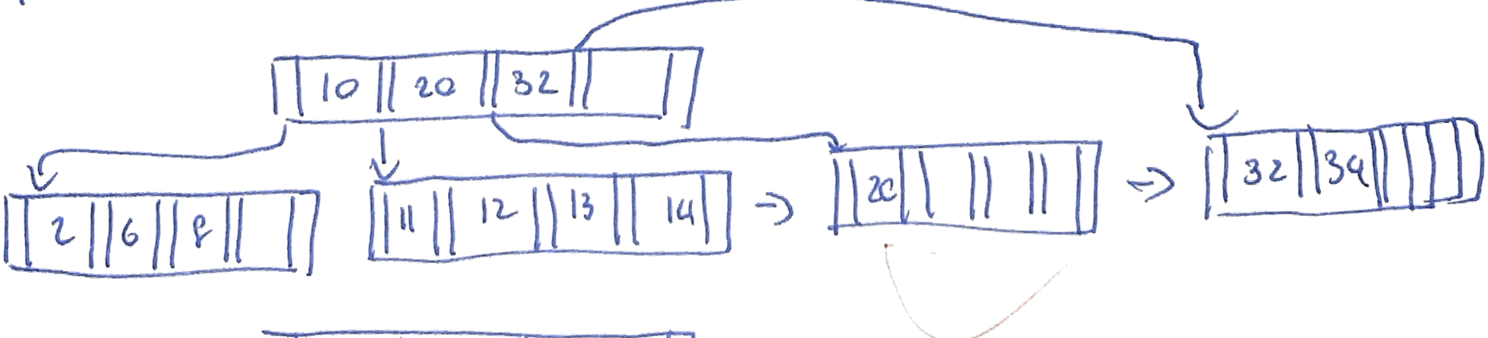
Part [B]

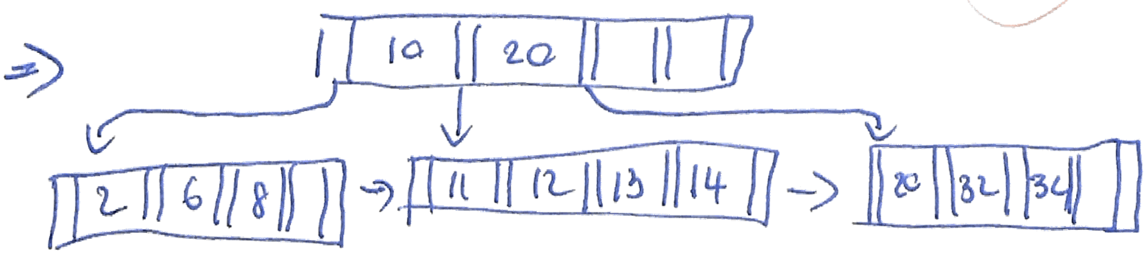


step 1: delete 26



step 2: delete 22





Problem 2

(21 points)

Assume that we have 100,000 tuples with only one 100-byte long attribute. Assume bucket size is 4096 bytes. Now we create an extendible hashing index on these tuples.

A (5 points)

How many buckets are in the hash index, and what is the global depth D, of the directory?

#number of tuple per bucket $= \frac{4096}{100} = 40$

$\Rightarrow$ # bucket $= \frac{100000}{40} = 2500$ bucket

~~The global depth D is the key size which map with directory which is 2500.~~

The global depth is ~~2500~~ 12.

B (8 points)

Assume that the global depth is D and the local depths for buckets are either D or $D-1$. Count the number of buckets with local depth D and those with local depth $D-1$.

total bucket we have in global .is

2^D

with local depth is D.

$\Rightarrow$ total bucket = ~~2~~ $2^D = 2^{12} = 4096$

-4

with local depth is D-1.

$\Rightarrow$ total bucket $= 2^D - 2^{D-1} = 2048.$

$= 2^{12} - 2^{11}$

C (8 points)

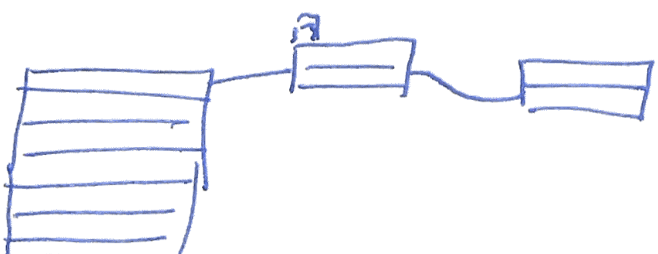
So, we have the situation described above and the fact that all the buckets are full, and now we insert a new record: will this insertion always cause an increase in the local depth of the bucket? And, will this always cause an increase in the global depth of the directory? Justify your answer.

After all the buckets are full.
if we insert a new record The local depth ~~will~~ ~~incresing~~ and ~~but~~ the plobal depth not changing.
they just add more bucket in the local bucket.

-2

for example

this bucket is full if we add more value to it will only add more bucket on local bucket



Problem 3

(62 points)

Given the following relations describing:

```
items (sku, iname, brand, desc)
stores (sid, sname, street_address, city, phone)
sells (sid, sku, price)
```

Assume the following:
(i) There are 100 pages for buffering in main memory.
(ii) Each page holds 4096 bytes; each attribute is 20 bytes long; each pointer in B+-tree takes 4 bytes.
(iii) The relation **items** has 100,000 tuples; the relation **stores** has 10,000 tuples, the relation **sells** has 1,000,000 tuples.
(iv) There are 10,000 item names and 1,000 brands in total. The price ranges from \$1 to \$1,000.
(v) The distribution of values is uniform.

Suppose that the following indexes exist on **items**: a primary B+-tree index on iname, and a hash index on brand.
Suppose that the following index exists on **sells**: a primary B+-tree index on sid, and a secondary B+-tree index on price.
There are no indexed on **stores**.

A (6 points) How many pages does each relation occupy on disk?

5

* items.

# of tuples/page $= \frac{4096}{20 * 4} = 51$

$\Rightarrow$ Number of pages $= \frac{1000000}{51} = 1960$ pages

* stores

# of tuples/page $= \frac{4096}{20 * 5} = 40.$

$\Rightarrow$ # of pages $= \frac{10000}{40} = 2500$ pages

250

* sells : # of tuple/page $= \frac{4096}{60} = 68$ $\Rightarrow$ total # of pages $= 14705$ pages

B (28 points) What is the minimal cost (in terms of numbers of pages transferred) of answering these queries?

(1) `SELECT * FROM items WHERE iname = "Headphone"`

using B+tree we have: # of pointer node $= \left\lceil \frac{4096-4}{20+4} \right\rceil + 1 = 171$

# of quality tuple $= 100000 * \frac{1}{10000} = 10$

height of B+ tree $= \left\lceil \log_{\frac{171}{2}} 10000 \right\rceil = 2.$

cost $= 10 + 2 = 12.$

4

(2) `SELECT * FROM items WHERE brand = "Beats"`

using the hash index

# of qualify tuple $= \frac{100000}{\cancel{1}000} = 100$

$\Rightarrow$ the cost $= 1.2 * 100 = 120$

7

(3) SELECT * FROM sells WHERE price = 980

using B+tree

height of B+tree = $\left\lceil \log_{\frac{171}{2}} 999 \right\rceil = 1$.

Number of qualify tuples = $\frac{1000000}{999} = 1001$.

5

cost = 1001 + 1 = 1002.

(4) SELECT * FROM sells WHERE price > 980

using B+tree

height of B+tree = $\left\lceil \log_{\frac{171}{2}} 999 \right\rceil = 1$.

Number of qualify tuples = $1000000 * \frac{(1000-980)}{(1000-1)}$

= 20020

cost = 20020 + 1 = 20021

3

Sequential file scan

(3) ~~Merge~~ Hash Join

if use stores as build relation. $= \log_{[M-1]} 2500 - 1$

if use sells as buil relation $= 1$.

3

$= \log_{99} 14705 = 2$.

* stores as build relation.

$c = 2 * (2500 + 14705) * 1 + 2500 + 14705$

$= 51615$

* sells as build relalation

$c = 2 * (2500 + 14705) * 2 + 2500 + 14705$

$= 86025$ ⇒ Minimum cost when stores is build relation.

(4) ~~Hash~~ Merge Join

$\text{cost} = 2\, br \left[\log_{99} \frac{br}{99}\right] + br$

1

* stores $= 2 * 2500 * \log_{99} \frac{2500}{100} + 2500$

$= 6002$.

* sells $= 2 * 14705 * \log_{99} \frac{14705}{100} + 14705$

$= 76121$.

total cost $= 6002 + 76121 + (2500 + 14705) * 2$

$= 116533$