b-trees 2.0
prof. Stratos Idreos

HTTP://DASLAB.SEAS.HARVARD.EDU/CLASSES/CS165/
HV Jagadish

Prof University of Michigan

10/6
Magdalena Balazinska
Prof University of Washington

10/13
10/12

**midterms**

how to prepare

- open book, notes, no laptop/discussion
- material from lectures only
- check all quizzes and questions

everything up to class 10 + 10/6 lecture

**quiz-like questions - no exact answer**

explain all steps and tradeoffs

speed testing - like

**expectations:** describe the design space - chose what you think is the best approach (>1 if we ask for it) and then analyze in detail all requests - if you made the wrong choice in the begging it is ok - but say so if you find out in the end and explain as much as possible

**Sunday before midterm:** Stratos office hours 2pm+
clustered (all columns)

index

data

secondary indexes
subset of columns
Btree on A, A is sorted, order is propagated to the rest of the columns

clustered index on A
(no need for positions)

every table can/should have one (be a) clustered index
Btree on C, copy of C is sorted, we keep a copy of the positions that map on the clustered index

secondary index on any column(s) needs positions
info to navigate lower level
value-pointer

<12  >=12

1,2,3…  12,15,17  20,…

sorted array

page size: 64K - holds 16K 4 byte ints
N elements, P pages

4+4 bytes for each page
(value+pointer)
64K/8= index 8K pages

N elements, P pages

can index 8K pages
of the next level

CS165, Fall 2016
Stratos Idreos
searching internal node

(v1,p1) (v2,p2) (v3,p3) (v4,p4) (v5,p5) (v6,p6)...

try binary search vs scan in your project
how big should nodes be?
the more data we can fit in a node (without changing its size) the faster our index becomes

(v1,p1) (v2,p2) (v3,p3)…
what does a leaf contain

searching leaf nodes

pointers or positions/ids
no perfect tree...

it depends on what we are trying to do
read/write ratio, data distribution,
response time guarantees
etc
updates

1,2,3…
12,20
30,50
50,…
35,…

12,15,17
20,…
...
updates

1,2,3…  12,15,17  20,…

2,5,6,3,2  22,25,24
updates

1,2,3... → 2,5,6,3,2

12,15,17 → 22,25,24

20,... → 7,8,3,5,4

30,50

12,20

35,...

50,...
updates

1,2,3... 12,15,17 20,...

2,5,6,3,2 22,25,24

7,8,3,5,4
good
new value
leaves

node capacity = 4
leaves

node capacity = 4
b-tree - dynamic tree - always balanced

every node/page is at least 50% full (except root)
node capacity = 4

insert "6"

node capacity = 4
delete "5"
node capacity = 4
how to load a b-tree

for(i=0; i<totalValues;i++)
    insertToBtree(tree,value[i])

?
buffers for loading

root level buffer

k level buffer

leaf level buffer

any problems
bulk loading
1. sort all
2. build tree

1,2  3,4  5,6  7,8  9,10
bulk loading

1. sort all
2. build tree
bulk loading
1. sort all
2. build tree
project milestone 2
(reasonably) cache conscious b-tree
use as both clustered and secondary index
(start with a linked list if no P0)
Employee(Id, name, address, office, salary, year hired, …)

We have a B-tree on table Employee which uses salary as the key and also contains attributes “name” and “year hired”.

We want to give a 5% raise to all employees that work for more than 10 years in the company and have a salary lower than 100K.

1) Write the SQL query
2) How to update the B-tree?
3) What is the query plan?
We want to give a 5% raise to all employees that work for more than 10 years in the company and have a salary lower than 100K.

```sql
update employee
set salary=salary*1.05
where salary<100K and year_hired<2006
```
b-tree on employee.salary

X

100K
b-tree on employee.salary

for each tuple if it qualifies (check years hired) update

100K
b-tree on employee.salary

for each tuple if it qualifies (check years hired) update

100K
b-tree on employee.salary

for each tuple if it qualifies (check years hired)
update

we are going to keep updating until everyone is at least at 100K
b-tree on employee.salary

get all qualifying IDs first then update in one go or maintain an extra structure e.g., a bit vector or hash on tuple ID to remember the updated tuples

for each tuple if it qualifies (check years hired) update

we are going to keep updating until everyone is at least at 100K
Modern B-Tree Techniques
by Goetz Graefe
Foundations and Trends in Databases, 2011

next up: guest lecture, midterm, guest lecture
index vs scan
b-trees 2.0
DATA SYSTEMS
prof. Stratos Idreos