Van Duong

CMPE 138

Homework 2

Problem 1: Disk Characteristics

(a) What is the total capacity of a track and what is its useful capacity (excluding interblock gaps)?
A track has 20 blocks. Each block has 512 bytes and an interblock gap of 128 bytes.
Total track capacity = (Block size + Gap size) × Blocks per track
= (512 + 128) × 20 = 12,800 bytes (12.8 KB)
Useful track capacity = Block size × Blocks per track
= 512 × 20 = 10,240 bytes (10 KB)

(b) How many cylinders are there?
A cylinder consists of all tracks at the same position on every disk surface.
There are 400 tracks per surface and 15 double-sided disks, meaning 30 surfaces.
Total cylinders = 400 (since a cylinder spans all surfaces but counts only per-track positioning).
= 400 cylinders

(c) What is the total capacity and the useful capacity of a cylinder?
Total cylinder capacity = Total track capacity × Number of surfaces
= 12,800 × 30 = 384,000 bytes (384 KB)
Useful cylinder capacity = Useful track capacity × Number of surfaces
= 10,240 × 30 = 307,200 bytes (307.2 KB)

(d) What is the total capacity and the useful capacity of a disk pack?
Total disk pack capacity = Total cylinder capacity × Number of cylinders
= 384,000 × 400 = 153,600,000 bytes (153.6 MB)
Useful disk pack capacity = Useful cylinder capacity × Number of cylinders
= 307,200 × 400 = 122,880,000 bytes (122.88 MB)

(e) Disk speed: Transfer rate, block transfer time, and average rotational delay
Revolutions per minute (RPM) = 2400
Revolutions per second = 2400 / 60 = 40
Time for one revolution = 1 / 40 = 25 ms
Block transfer time (BTT) = Time per track / Number of blocks per track
= 25 / 20 = 1.25 ms per block
Transfer rate = Total useful track capacity / Time per track

= 10,240 bytes / 25 ms = 409.6 bytes/ms
Average rotational delay (RD) = ½ of one full revolution
= 12.5 ms
Bulk transfer rate = Total track capacity / Time per track
= 12,800 / 25 = 512 bytes/ms

(f) Average time to locate and transfer a block
Seek time = 30 ms
Rotational delay = 12.5 ms
Block transfer time = 1.25 ms
Total average time = Seek time + Rotational delay + BTT
= 30 + 12.5 + 1.25 = 43.75 ms

(g) Time to transfer 20 random blocks vs. 20 consecutive blocks using double buffering
For random blocks: Each block requires a seek, rotational delay, and transfer time
= 20 × 43.75 = 875 ms
For consecutive blocks with double buffering:
First block takes 43.75 ms
Next 19 blocks only take block transfer time
= 43.75 + (19 × 1.25) = 67.5 ms

Problem 2: B+ Tree Index

(a) How many pages can we store in the first (root) level? Second level? Tenth level?
Fan-out (f) = 30
Root level (L1) = 1 pages
Second level (L2) = 30^2 = 30 pages
Tenth level (L10) = 30^9 pages

(b) Levels required and space usage per level

Number of rows = 2 billion (2×10^9)

Number of level = log_f(N) = log_30(2billion) = 6.29 (round up to 7)
(where f is fan-out, N is number of rows)
=> so we will have 7 levels

Now we need to calculate number of space required:

- Number of leaf page = 2*10^9/30 = 66.7 million

Level	Approx Pages (rounded up)	Calculation
L1 (Leaf)	66,667,000	2,000,000,000 / 30
L2	2,222,234	66,667,000 / 30
L3	74,075	2,222,234 / 30
L4	2,470	74,075 / 30
L5	83	2,470 / 30
L6	3	83 / 30
L7 (Root)	1	3/30

- Then we can calculate Space Required per Level = Number_of_Page *4 / 30

Level	Pages	Space (MB)	Rounded
L1	66,667,000	(66,667,000 × 4) / 1024 = 260,582.0	260,582.0 MB
L2	2,222,234	8,679.80	8,679.8 MB
L3	74,075	289.3	289.3 MB
L4	2,470	9.6	9.6 MB
L5	83	0.3	0.3 MB
L6	3	0.01	0.01 MB
L7	1	0.004	0.01 MB

(c) Worst-case IO (disk accesses) to find a record

Disk accesses per level = 1
Total levels = 7
Worst-case IO = 7 disk accesses

Problem 3: Join Costs

a) IO = P(R)+T(R)*P(S)+P(R,S) = 20+1,600*200 +100 = 320,120

b) IO = P(S) + T(S)*P(R)+P(R,S) = 200+15,000*20 +100 = 300,300

c) IO(R,S) = P(R)+P(R)*P(S)/B+P(R,S) =20+20*200/32+100=245
IO(RS,T) = P(R,S) + P(R,S)*P(T)/B +P(R,S,T) = 100+100*2000/32+500 = 6850
IO(S,T) = P(S)+P(S)*P(T)/B+P(S,T) = 200+200*2000/32+1000 = 13600
IO(ST,R) = P(S,T) +P(S,T)*P(R)/B+P(S,T,R) = 1000+1000*20/32+500 = 2125

d) IO(R,S) = ~Sort(P(R)) + Sort(P(S)) + P(R) + P(S) = 40 + 800 + 20 + 200 = 1060
IO(RS,T) = ~Sort(P(R,S)) + Sort(P(T)) + P(R,S) + P(T) = 400 + 8000 + 100 + 2000 = 10500

e) IO(S,T) = ~Sort(P(S)) + Sort(P(T)) + P(S) + P(T) = 800 + 8000 + 200 + 2000 = 11000
IO(ST,R) = ~Sort(P(S,T)) + Sort(P(R)) + P(S,T) + P(R) = 4000 + 40 + 1000 + 20 = 5060

f) IO(R,S) = ~Sort(P(R)) + Sort(P(S)) + P(R) + P(S) = 40 + 800 + 20 + 200 = 1060

g) IO(R,S) = 2 * (P(R) + P(S)) + (P(R) + P(S)) = 2 * (20 + 200) + (20 + 200) = 660

IO(RS,T) = 2 * (P(R,S) + P(T)) + (P(R,S) + P(T)) = 2 * (100 + 2000) + (100 + 2000) = 6300

h) IO(S,T) = 2 * (P(S) + P(T)) + (P(S) + P(T)) = 2 * (200 + 2000) + (20 + 2000) = 6420

IO(ST,R) = 2 * (P(S,T) + P(R)) + (P(S,T) + P(R)) = 2 * (1000 + 20) + (1000 + 20) = 3060

Problem 4: Linear Hashing

(a) Buckets in primary area
Load factor = 0.7
Total records = 112,000
Records per bucket = 20
Buckets needed = 112,000 / (20 × 0.7) = 8000 buckets

(b) Number of bits for bucket addresses
Buckets = 8000
Bits required = log_2(8000) ≈ 13 bits