ADBT, JUNE 6, 2005

PROBLEM 1

a) WITH THE GIVEN ASSUMPTIONS, SCENARIO 1 HAS NO SPACE OVERHEAD.

THE SPACE FOR THE B-TREE INDEX IS DOMINATED BY THE LEAVES, WHICH USE (ESSENTIALLY) $8+4=12$ BYTES PER KEY, I.E. $12N$ BYTES IN TOTAL.

b)
- THE BOUND FOR INSERTION IN A HASH TABLE IS $O(1)$ I/Os, AND IN A B-TREE $O(\log_B N)$ I/Os. (DOMINATES $O(1)$ I/Os.)
- IN BOTH CASES, A POINT QUERY CAN BE DONE USING THE HASH TABLE IN $O(1)$ I/Os.
- A RANGE QUERY CAN BE IMPLEMENTED AS r POINT QUERIES IN $O(r)$ I/Os, OR AS A RANGE QUERY IN ~~THE~~ B-TREE IN $O(\log_B N + \ell/B)$ TIME, PLUS $O(\ell)$ TIME FOR FOLLOWING POINTERS.

c) SCENARIO 1 IS BEST WHEN:

$$X + Y + Zr < X \log_B N + Y + Z(\log_B N + \ell)$$
$$\Updownarrow$$
$$Z(r - \log_B N - \ell) < X(\log_B N - 1)$$

PROBLEM 2

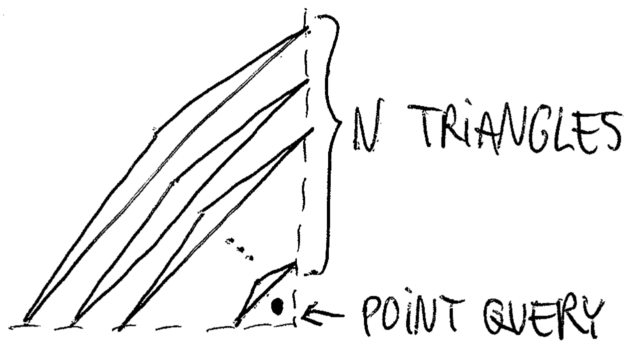
a) R1, R2, R4, A, B, R5, R3, R6, R7

MATCH

UNLESS SEARCH ALG. KNOWS THERE IS AT MOST

b) THIS IS ESSENTIALLY THE "PLANAR POINT LOCATION" PROBLEM. THE ONLY DIFFERENCE IS THAT WE DISCUSSED THAT PROBLEM FOR A POLYGON SPLIT IN PARTS USING LINE SEGMENTS. WE CAN REDUCE THIS CASE TO THAT ONE BY ARBITRARILY CONNECTING THE TRIANGLES WITH LINES. THIS GIVES A SOLUTION WITH QUERIES IN $O(\log_B N)$ I/Os.

c) CONSIDER THE FOLLOWING EXAMPLE:



SINCE THE BOUNDING BOX FOR ANY SUBSET OF THE TRIANGLES CONTAINS THE POINT, A SEARCH MUST VISIT ALL NODES IN THE R-TREE TO SEE THAT THE POINT IS IN NO TRIANGLE.

PROBLEM 3

a)

	EST. $\|R_1 \bowtie R_2\|$	EST. $\|R_2 \bowtie R_3\|$
SIGNATURE 1	900,000	700,000
SIGNATURE 2	1,100,000	1,050,000
SIGNATURE 3	630,000	1,980,000

$|R_1 \bowtie R_3| = |R_1 \times R_3| = |R_1|\,|R_3|$, SO NO ESTIMATE IS NEEDED.

b)

EST. $|R_1 \bowtie R_2|$: $100 \cdot 100^2 = 1{,}000{,}000$ ← 12 BYTES / TUPLE: 12,000,000 BYTES

EST. $|R_2 \bowtie R_3|$: $200 \cdot 100^2 = 2{,}000{,}000$ ← 12 BYTES / TUPLE: 24,000,000 BYTES

EST. $|R_1 \bowtie R_3|$: $2000 \cdot 100^2 = 20{,}000{,}000$ ← 16 BYTES / TUPLE: 320,000,000 BYTES

c)

- SINCE THE JOIN ALGORITHM IS LINEAR IN THE SIZE OF ITS INPUT AND OUTPUT, THE TOTAL COMPLEXITY IS LINEAR IN THE SIZE OF THE RELATIONS AND FINAL RESULT (FIXED), PLUS THE SIZE OF THE INTERMEDIATE RESULTS. HENCE IT IS OPTIMAL TO MINIMIZE INTERMEDIATE RESULTS.
- IF WE TAKE 4 RELATIONS $R_1, .., R_4$ WITH N TUPLES EACH WITH NO COMMON ATTRIBUTES AND COMPUTE THE NATURAL JOIN (CARTESIAN PRODUCT) USING A LEFT-DEEP JOIN TREE THERE IS AN INTERMEDIATE RESULT OF SIZE N^3. ON THE OTHER HAND, THIS EXPRESSION ONLY HAS INTERM. RESULTS OF SIZE $\approx 2N^2$: $(R_1 \bowtie R_2) \bowtie (R_3 \bowtie R_4)$.

PROBLEM 4

a)

LOCK ON	OBTAINED BEFORE	RELEASED AFTER
FREEBUSINESSSEATS (X)	1	7
UPGRADES (X)	2	6
SEATS (X)	3	7

b) TO MINIMIZE LOCKING TIME, AT LEVEL "READ COMMITTED" THE TRANSACTION WOULD CARRY OUT 4 AND 5 WITH A SHARED LOCK ON Seats WHICH IS RELEASED. THUS, TRANSACTION 1 MAY OBTAIN AN EXCLUSIVE LOCK ON Seats UP TO 6 WHERE TRANSACTION 2 REQUESTS AN EXCLUSIVE LOCK.

c)
- IF TRANSACTION 1 HAS THE SMALLEST TIMESTAMP, IT IS ROLLED BACK WHEN TRYING TO UPDATE Seats, BECAUSE IT HAS A LARGER READ TIME STAMP.
- IF TRANSACTION 1 HAS THE LARGEST TIMESTAMP, IT WILL WAIT FOR T2 TO COMMIT/ABORT BEFORE IT CAN UPDATE. LATER, WHEN T2 WANTS TO UPDATE WHAT T1 HAS READ, IT WILL BE ROLLED BACK AND T1 WILL COMPLETE.

PROBLEM 5

a) • ITEMS FOUND IN FIRST PASS: milk, diapers, eggs, beer, chips

• COUNT MATRIX FOR SECOND PASS:

	milk	diapers	eggs	beer	chips
diapers	3				
eggs	2	3			
beer	1	1	0		
chips	0	1	0	3	

FOUND PAIRS ARE {milk, diapers}, {eggs, diapers} AND {beer, chips}.

b)

	INTEREST	CONFIDENCE
milk → diapers	$3/4 - 5/8 = 1/8$	$3/4$
diapers → milk	$3/5 - 4/8 = 1/10$	$3/5$
eggs → diapers	$3/3 - 5/8 = 3/8$	$3/3$
diapers → eggs	$3/5 - 3/8 = 9/40$	$3/5$
beer → chips	$3/4 - 3/8 = 3/8$	$3/4$
chips → beer	$3/3 - 4/8 = 1/2$	$3/3$

c) FOR THE FIRST PASS, SORT ALL ITEMS ANNOTATED WITH BUCKET NUMBERS SO THAT THEY CAN BE COUNTED EASILY: $O(\frac{N}{B} \log_{M/B}(N/M))$ I/Os.

NOW SORT THE HIGH SUPPORT ITEMS BY THEIR BUCKET NUMBERS – THIS GIVES THE ORIGINAL BUCKETS WITH LOW SUPPORT ITEMS REMOVED. FOR EACH BUCKET, GENERATE ALL PAIRS OF ITEMS, AND SORT THE PAIRS OF ALL BUCKETS SUCH THAT IDENTICAL PAIRS BECOME ADJACENT. FINDING HIGH-SUPPORT ITEMS CAN NOW BE DONE EASILY IN A SINGLE SCAN.