JUL THURSDAY 211-154 Wk 31 — 30

2015

Assignment - 4 (OS)

(06) Distributed Deadlock

S1 : P1 → P2, P3 → P4
S2 : P2 → P5, P5 → P6
S3 : P6 → P1

a) Combine all edges from the three sites:
P1 → P2, P2 → P5, P5 → P6, P6 → P1, P3 → P4
Processes P3 & P4 form an unrelated waiting pair, they are not part of the other chain.

b) Deadlock detection:
Yes, deadlock exists here - there is a cycle:
P1 → P2 → P5 → P6 → P1
This is a directed cycle. So processes involved in the deadlock are: P1, P2, P5, P6

P3 and P4 aren't part of the deadlock (they form a separate single wait edge P3 → P4 only)

(7) Distributed 0.3

a) Probability local = 1 − 0.3 = 0.7
0.7 × 5ms = 3.5ms.
Probability remote = 0.3
0.3 × 25ms = 7.5ms.
Sum: 3.5 ms + 7.5ms = 11.0ms
Expected access time = 11ms.

b) Strategy: Client-side read cache w/ server

Leases (LRU replacement).
Why → caching turns remote accesses into fast local reads, cutting avg latency.
Leases let clients safely use cached data w/o contacting the server on every read.
LRU is a simple, effective eviction policy.

Q8) a) Proposed Schedule (10 second window)
- Do one full checkpoint at the start (1,0): 200ms
- Do cremental checkpoints every 1 sec afterwards at t = 1, 2, 3 ---- 9 (9 incremental x 50ms = 450ms)
Total checkpoint overhead in 10 sec = 200 + 450 = 650 ms
Avg overhead = 650 ms / 10s = 65 ms per sec.

b) Reason:
→ RPO = 1 sec = at most 1 sec of lost work.
Doing an incremental every 1 sec guarantees that in the worst case, we can recover within 1 sec.

→ Incremental checkpoints are cheap, so using them frequently is the low cost way to meet RPO.

→ Full checkpoints are spokne since, so minimizing their frequency. One day full at the start provides the required baseline for replay

the chain of incrementals.
→ This min therefore satisfies the RPO while minimizing total checkpoints time in the 10sec window.

Q9) a) Sudden traffic spikes create hotspots and overload specific servers.
→ Maintaining low latency while balancing load across distant regions.
→ Keeping session/cart state local while still distributing load.
→ Avoiding cascading failures when one region becomes saturated.
Suitable algorithm ÷ Power of Two choices (P2C) load balancing bcoz it spreads load evenly and reduces hot spots.

b) ⇒ For low RPO:
Replicate critical data to other region continuously asynchronous replication gi small RPO, synchronous replication guess near-zero RPO).

→ For low RTO:
Use a global load balancer that quickly reroutes traffic when a region fails, plus automated failover to promote replica

SUNDAY 02