DDBS HW3

Massimo Hong

2020280082

1.

1.1

S1 = W2(x), W1(x), R3(x), R1(x), W2(y), R3(y), R3(z), R2(z)

S2 = R3(z), R3(y), W2(y), R2(z), W1(x), R3(x), W2(x), R1(x)

S3 = R3(z), W2(x), W2(y), R1(x), R3(x), R2(z), R3(y), W1(x)

S4 = R2(z), W2(x), W2(y), W1(x), R1(x), R3(x), R3(z), R3(y)

Conflicting operations in S1:

- W2(x) → W1(x)
- W2(x) → R3(x)
- W2(x) → R1(x)
- W1(x) → R3(x)
- W2(y) → R3(y)

Conflicting operations in S2:

- W2(x) ← W1(x)
- W2(x) ← R3(x)
- W2(x) → R1(x)
- W1(x) → R3(x)
- W2(y) ← R3(y)

Conflicting operations in S3:

- W2(x) → W1(x)
- W2(x) → R3(x)
- W2(x) → R1(x)
- W1(x) ← R3(x)
- W2(y) → R3(y)

Conflicting operations in S4:

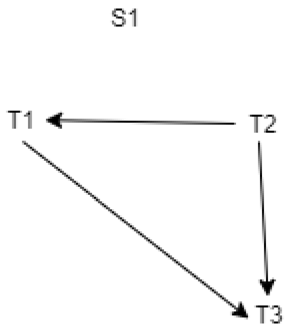
- W2(x) → W1(x)

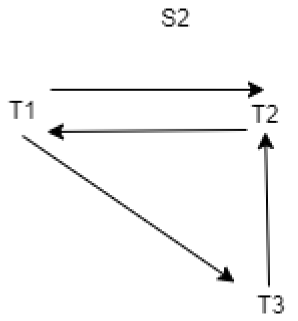
- W2(x) → R3(x)
- W2(x) → R1(x)
- W1(x) → R3(x)
- W2(y) → R3(y)

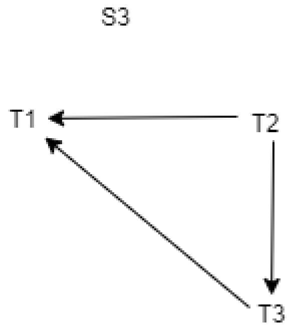
We can notice that S1 and S4 have the exact same conflicting operations, so we conclude that they are conflict equivalent.

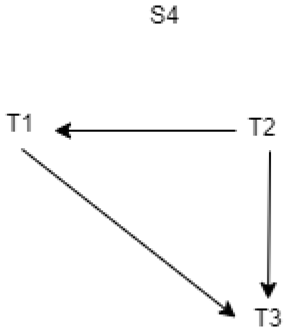
1.2

By looking at the conflicting operations found in 1.1, we can get the following transaction serialization graphs:









As we can see, only S2 has a cycle, so s2 is not serializable.

S1 can become: T2 → T1 → T3

S3 can become: T2 → T3 → T1

S4 can become: T2 → T1 → T3

2.

Centralized approach: a central component has the responsibility to detect deadlocks.

Advantages:

- low cost
- easy to implement
- better for DBMS with uniform access patterns across sites.

Disadvantages:

- high workload on the responsible node, frequent communication
- if the responsible node reaches failure, the entire system will collapse

Hierarchical approach: as the name suggests, the sites are organized into a hierarchical structure. A handful of responsible nodes are selected and each of them detects the descendant sites.

Advantages:

- The responsibility is well partitioned
- Less risk of collapse of the entire system
- Less dependence on a single node
- More efficient for localized access patterns as its deadlock detection functionality is directly proportional to the probability of deadlock occurrence.

Disadvantages:

- Less efficient in case a deadlock involves several clusters
- Harder to implement
- Higher maintenance cost

3.

- If there is a cycle that does not involve T_ex, we can simply conclude that the deadlock is local.
- It there is a cycle involving T_ex, then it is possible that exists a multi-site deadlock or even a global deadlock.

4.

We are guaranteed to find a global deadlock if it happened before the first WFG had been sent. With this method we can recover a deadlock before a node receives a graph with its own nodes.