CS5280 : ASSIGNMENT 2

Raj Patil : CS18BTECH11039

2021-10-02 Sat 16:41

Contents

Concurrency Control in Transactional systems
Theory Assignment 2

1 Ex 3.8

Using the definition obtained by the Herbrand Semantics of view seriazability, each read operation should be preceded by the same write for all the equivalent serial schedules. Now for a schedule $s \in VSR \setminus CSR$. The conflict graph has a cycle but the reads-from relation (i.e. view semantics) matches that of a serial schedule. This means that the reading semantics are consistent with a serial schedule and the conflict arises due to differing Herbrand semantics of some write. As this write (conflicting operation) did not change the view-semantics implies that it must not have any reads before i.e. it is blind. This follows as such writes do not change the view of the system as they do not update any previously read value (the semantics of these reads will remain unchanged). Hence such an s wil have blind writes. Consequently for s without any blind writes $CSR = VSR$.

2 OCSR Graph Characterization

Given

$$s = r_1(x)w_1(z)r_2(z)w_1(y)c_1r_3(y)w_2(z)c_2w_3(x)w_3(y)c_3$$

recalling the answer given in the quiz

2.1 Proposed characterization

For a graph characterization of OCSR, one can naturally extend the graph characterization of CSR as follows:

building upon the basics of CSR let $G = (V, E)$ be the base graph where $V = trans(s)$ and E is an augmented edge set . . .

given $s \in OCSR \to s \in CSR$, let $s' \approx_c s$ be the corresponding serial history

Now let $E = E_1 \cup E_2$ where E_1 corresponds to the edge set of the corresponding CSR graph.

$$E_1 = \{(t_i, t_j) : \exists p \in op(t_i), \exists q \in op(t_j)((p, q) \in conf(s))\}$$

Now to introduce cyclicity in case of failure, we precisely add edges by defining E_2 such that a cycle is introduced when $s \notin OCSR$

$E_2 = \{(t_i, t_j) : \neg(\forall s', s \approx_c s', \forall p \in op(t_i), \forall q \in t_j (p <_s q) \to \forall p \in t_i, \forall q \in t_j (p <_{s'} q))\}$

To understand this, see that if this favourable condition(note the outermost negation) is not followed, an edge is introduced in the graph that goes against the topological sort in the CSR graph and a cycle is introduced. Also note that this edge is introduced only if this happens for all the conflict equivalent serial histories s' and not just for any single history.

Hence formalizing $OG(s)$:

$$OG(s) = (V, E) \ such \ that \ V = trans(s), E = E_1 \cup E_2$$

where E_1 and E_2 are as defined above

Note that this is efficient (can be done in polynomial time), as one can obtain the equivalent histories efficiently from the initial topological sort and latter edge addtions can be computed in polynomial time as well. Finally cycle detection for a graph can be done by polynomially bounded algorithms as well, hence the characterization over all is efficient.

2.2 Application

The base CSR graph will look like

$$t_2 \leftarrow t_1 \rightarrow t_3$$

The corresponding set of conflict equivalent serial histories for this would be t_1, t_2, t_3 or t_1, t_3, t_2

To see if any new edges need to be added to this graph, checking as proposed..

- see that only transaction level order in s is $t_1 <_s t_3$
- there exists a serial history where this is true (pick any one of the two)
- hence the negation of the corresponding predicate will be False and no edges need to be added that could have disturbed the acyclic nature of the conflict graph.

Hence, the give schedule s is in $OCSR$ as the corresponding graph

$$t_2 \leftarrow t_1 \rightarrow t_3$$

is acyclic