Validation Based Protocol

Validation Based Protocol is also called Optimistic Concurrency Control Technique.

This protocol is used in DBMS (Database Management System) for avoiding concurrency in transactions.

It is called optimistic because of the assumption it makes, i.e. very less interference occurs, and therefore, there is no need for checking while the transaction is executed.

All updates are applied to local copies of data items kept for the transaction. At the end of transaction execution, while execution of the transaction, a **validation phase** checks whether any of transaction updates violate serializability. If there is no violation of serializability the transaction is committed and the database is updated; or else, the transaction is updated and then restarted.

Optimistic Concurrency Control is a three-phase protocol. The three phases for validation based protocol:

1. **Read Phase:**
   Values of committed data items from the database can be read by a transaction. Updates are only applied to local data versions.

2. **Validation Phase:**
   Checking is performed to make sure that there is no violation of serializability when the transaction updates are applied to the database.

3. **Write Phase:**
   On the success of the validation phase, the transaction updates are applied to the database, otherwise, the updates are discarded and the transaction is slowed down.

The **idea behind optimistic concurrency** is to do **all the checks at once**; hence transaction execution proceeds with a **minimum of overhead** until the validation phase is reached. These circumstances are not much favorable for optimization techniques.

Validation based protocol is **useful for rare conflicts**. Since only **local copies** of data are included in rollbacks, **cascading rollbacks are avoided**.

This method is **not favorable for longer transactions** because they are **more likely** to **have conflicts** and might be repeatedly rolled back due to conflicts with short transactions.

Following three time-stamps that we assigned to transaction T_i, to check its validity:

1. Start(Ti): It is the time when Ti started its execution.

2. Validation(Ti): It is the time when Ti just finished its read phase and begin its validation phase.

3. Finish(Ti): the time when Ti end it’s all writing operations in the database under write-phase.

Two more terms that we need to know are:

1. **Write_set**: of a transaction contains all the write operations that Ti performs.

2. **Read_set**: of a transaction contains all the read operations that Ti performs.

In the Validation phase for transaction Ti the protocol inspect that Ti doesn’t overlap or intervene with any other transactions currently in their validation phase or in committed.

The validation phase for **Ti** checks that for **all transaction Tj** one of the following below conditions must hold to being validated or pass validation phase:

1. Finish(Tj)<Starts(Ti), since Tj finishes its execution means completes its write-phase before Ti started its execution(read-phase). Then the serializability indeed maintained.

2. Ti begins its write phase after **Tj** completes its write phase, and the read_set of **Ti** should be disjoint with write_set of **Tj**.

3. Tj completes its read phase before **Ti** completes its read phase and both read_set and write_set of **Ti** are disjoint with the write_set of **Tj**.