**Proof that Undo/Redo satisfies Undo and Redo rules**

The protocol applies *undo* and *redo* operations to recover the database from a failure. A successful recovery by the protocol requires that each transaction during execution must satisfy undo and redo rules. *It is required to prove that if these rules are not followed by transactions then this protocol cannot recover the database from failure*, i.e., *the protocol satisfies these rules.*

**Proof**

Let T = {t1, t2, …, tm} is a set of transactions executing concurrently under a CCM.

Let D = {d1, d2, …, dn} is a database composed of data items di’s.

Let L = {l1, l2, …, lp} is a log on a stable storage (log disk) containing records li’s for a set of transactions (ti ∈ T).

Let C = {c1, c2, …, cq} is the cache containing records ci’s.

A ti ∈T processes di (BFIM)∈D and generates d’i (AFIM) ∈ C. For undoing ti ∈ T, di (BFIM) must be available and for redoing d’i (AFIM) must be available to this protocol. The RM (Recovery Manager) requires that every transaction must follow WAL (Write Ahead Logging) protocol. The WAL guarantees that availability of di (BFIM) and d’i (AFIM) are available in li ∈ L and l’i ∈ L (this satisfies undo and redo rules).

Suppose undo/redo protocol does not satisfy undo and redo rules. This implies that at any time during execution, ti has no log record for the data item di it processed (i.e., li ∉L) which guarantees that di (BFIM) ∉L. This will happen iff ti does not follow WAL which means the protocol violates undo rule. Further, if d’i (AFIM)∉L and if d’i(AFIM) is in D then this violates WAL, that is ti did not follow WAL. This establishes that our assumption that undo/redo protocol does not satisfy undo and redo rules is false.