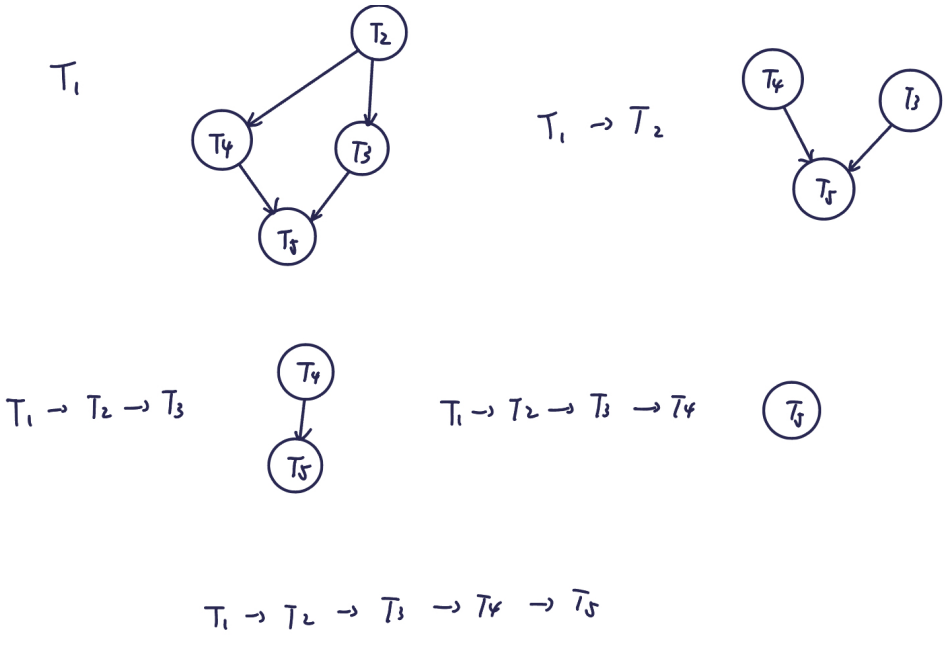
DB - HW13

17.6

Consider the precedence graph of Figure 17.16. Is the corresponding schedule conflict serializable? Explain your answer.

Answer:

There is a serializable schedule corresponding to the Figure 17.16, for we can obtain a possible schedule by doing a topological sort by the way shown in the Figure below.



Clearly, $T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_4 \rightarrow T_5$ is a possible sequence.

17.7

What is a cascadeless schedule? Why is cascadelessness of schedules desirable? Are there circumstances under which it would be desirable to allow noncascadeless schedules? Explain your answer.

Answer:

A cascadeless schedule is a schedule where, for each pair of transactions T_i and T_j that T_j reads data previously written by T_i , the commitment operation of T_i always appears before the read operation of T_j.

The cascadelessness of schedule is desirable for, in cascadeless schedules, the failure of a tracnsaction would not cause the aborting of any other transaction.

However, when the probability of failure is very low, the price of the cascading aborts is acceptable, and the concurrency would be increased. Under this circumstances, noncascadeless schedule might be desirable.

17.12

List the ACID properties. Explain the usefulness of each.

Answer:

- Atomicity

  For all the operations of a certain transaction should either be reflected properly in the database, or none of them were, the lack of atomicity would cause the inconsistency in the database.

- Consistency

  Execute the transactions non-concurrently preserves the consistency. This property insures that the transactions must transform the database from one state of consistency to another.

- Isolation

  Isolation means, when multiple transactions are executed concurrently, for every pair of transactions T_i and T_j , it appears to T_i that either T_j finished execution before itself started, or T_j started execution after T_i finished. Thus, each transaction is unaware of other transactions executing concurrently with it.

  Isolation is required for the user view of a transaction system needs it. Besides, it ensures that:

  - During the execution of a transaction, the intermediate (and possibly inconsistent) state of the data should not be exposed to all other transactions.
  - Two concurrent transactions should not be able to manipulate the same piece of data.

- Durability

  Durability means after a transaction is successfully completed, the changes it made to the database would persist perpetual, even when the system fails. It makes the data recoverable.