

16.27

Deadlock avoidance is more preferable in scenarios where deadlocks are very likely to happen. Remedying a deadlock requires rolling back a transaction, and thus, wastes all of its computation. Therefore, in environments with frequent deadlocks, deadlock remedying entails expensive overhead.

Why would deadlocks happen frequently? Typical reasons include: too many transactions, each transaction needs substantial I/O accesses, or there are some hot-spot data that many transactions need to read and modify.

16.28

Yes starvation may still happen. This is clear in the 2PL variation that avoids deadlocks (we discussed this variation in class). Recall that, under this protocol, a transaction must, at the very beginning, acquire all the locks on the items it needs to read/modify; otherwise, it must relinquish all the locks and wait. Hence, it is not easy for a transaction, which needs to access a large number of data items, to obtain all the locks. In this case, the transaction may need to wait a long time, i.e., it starves.