**What is indefinite postpone? How it’s different from deadlock**

When a thread waits indefinitely for other threads to be interrupted so that it can complete its task, this situation is known as indefinite postponement. The thread in a wait state is "deprived" of the opportunity to change to a "runnable" state due to variables like higher priorities on OS tasks or threads.

Deadlock and indefinite postponement are different

When a process asks for a resource that won't become available, it is said to be in a deadlock. If a process is frequently delayed over an extended length of time while the system is focusing on other processes, i.e., the process is ready to move forward but is never given the CPU, it is said to be endlessly postponed.

**Why is deadlock prevention not a primary function for many operating system?**

A deadlock occurs when two computer programs that are using the same resource effectively block each other from using it, which causes both programs to stop working. One program could only run at a time on the first computer operating systems.

For the majority of personal computer users, who are frequently more concerned with operating system performance and feature, deadlock is a rare and sometimes dismissed minor nuisance.

**Why might deadlock in distributed system be more difficult to detect than in a single computer**

It is a challenging task to identify generalized deadlocks in distributed systems since this requires identifying a complex topology in the global WFG. A process can only have one open request for one unit of a resource in the single resource paradigm.

Methods for spotting stalemate in a distributed system

However, the drawbacks include single-point failure and excessive strain on a single node (i.e., the entire system is dependent on one node, and if that node fails, the entire system crashes), which reduce the system's dependability.

**(T/F) A system can eliminate deadlock by choosing at random an irreducible process in a resource allocation graph**

Answer: False

By the time a deadlock is discovered, the system may already have several circular waits.

**Why might a system that restarts a process it "kills" to break a deadlock suffor from poor performance?**

Firstly, since stopping a process results in lost work.

Second, because the restart procedure will run the same code that resulted in the initial deadlock, the system may repeatedly enter a deadlock if no changes are made.

**Why might deadlock detection can be a better policy than either deadlock prevention or deadlock avoidance? Why might it be a worse policy?**

In general, resource allocation restrictions are lessened by deadlock detection, increasing resource utilization. However, it necessitates the execution of the deadlock detection algorithm on a frequent basis, which might result in significant cost.

**Suppose a system attempt to reduce deadlock detection overhead by performing deadlock detection only when there are large number of process in the system. What is one drawback to this strategy?**

If there are only a few processes in the system, the system may not be able to identify certain deadlock because it can happen between two processes.