In order for mutual exclusion to be satisfied in a system, a critical resource granted to a process must be released before it can be granted to another process. There are two conditions for distributed mutual exclusion. First, requests for a resource should be granted in the order in which they were made. Second, every granted resource must eventually be released, to ensure every request will eventually be granted.

In the proposed algorithm, mutual exclusion is primarily granted via the use of a single token in the distributed system. In order to enter critical section, a node must become the holder of the token.

Requests are granted in the order they are made through the use of a first in first out queue of requests. Starvation is a condition that occurs when one process must wait indefinitely to enter its critical section even though other nodes are entering and exiting their own critical sections.

The first in first out queue system for requests prevents a starvation condition, by ensuring that all processes are eventually granted as they progress in the queue. Furthermore, all processes are treated fairly because of the lack of prioritization for requests. In addition to the use of the queue, a sequence number per node is used so that nodes can track which messages are being responded to.

The proposed algorithm also uses a quorum based organization structure, which further ensures fairness for all processes by adhering to the non-null intersection property to ensure that processes are all located relatively close to the token holder, regardless of who the token holder is.

Deadlock is a condition that occurs when no process in the system is in its critical section and no requesting process can ever proceed to its own critical section. The use of a single token prevents deadlock from occurring in the proposed algorithm. The holder of the token will always grant another new process the token if a request is made, regardless of the number or nature of requests.