- Suzuki-Kasami Broadcast Algorithm is a token based distributed mutual exclusion Algorithm designed to ensure that only one Process at a time enters the Critical Section in a distributed system or distributed DB Environment.

It is especially suitable for systems where read/write operation on shared databases must be synchronized without using a Centralized Coordinator.

In Distributed Systems,

- multiple nodes may attempt simultaneous updates.
- Concurrency can cause inconsistency.
- mutual exclusion ensures serializability.

Suzuki-Kasami improves over permission-based algorithms by:

- Reducing message Complexity
- Concurrency can cause inconsistency
- mutual exclusion ensures serializability
- Eliminating deadlocks.
- Avoiding starvation.

⇒ Datastructures Used ⇒ Request Number (RN) Array
RN[i] = largest request number from P_i to this Process

b). Token (Unique, Only One Exist)

- $LN[i]$: Last request number from Process P_i that has been Satisfied.
- Q: FIFO queue of requesting Process.

<u>Algorithm Operation</u>

Step ①: Requesting the Critical Section: when Process P_i wants to Enter the CS.
- Increments its request number.
  $RN[i]++;$
- Broadcast REQUEST $(i, RN[i])$ to all Other.

Step ②: when Process P_j receives a REQUEST (i, n)

$RN[i] = MAX(RN[i], n)$

If P_j holds the token and:

$RN[i] = LN[i] + 1$

Then P_i is eligible → token may be sent.

Step ③: Token Transfer:

If the token header
- Is not in CS

Find the pending request.

Then,
① Add requesting Process to token queue.
② Send token to first Process in Q.

Step ④: Entering the Critical Section: If P_i receive token it Enters

Step ⑤ : Releasing the Critical Section.

After Exiting CS:

(i) Update :

$LN[i] = RN[i]$

(2) Checking for other pending requests.

If $RN[j] = LN[j]+1 \Rightarrow$ add j to Q.

(3) If queue not empty ⇒ Pass token.

- Mutual Exclusion : Only the token holder can enter CS.
- Deadlock freedom : Token circulation ensures progress.
- Starvation Freedom : FIFO queue ensures fairness.

Message Complexity

Request : N-1 messages (broadcast)

Token transfer : 1 message.

Total per CS Entry : O(n)

Applications in Distributed System

- Distributed transaction management
- Replicated database updates
- Distributed file systems.
- Shared resource coordination

Advantages

- No deadlock
- No starvation
- Fair access (FIFO)
- Lower message overhead than Permission - based algorithm
- Efficient when CS access is frequent.

Disadvantages

- Token loss causes failure
- Not fault tolerance
- Broadcast is high for largest system
- Assume reliable Communication

Suzuki-Kasami Broadcast Algorithm Provides an efficient, fair and deadlock - free sol^n for mutual exclusion in distributed databases. Its token based approach reduces Communication Overhead and ensures strict access control, making it ideal for medium - Scale distributed systems where relaibility is high.