Token-Based Algorithms

- In token-based algorithms, a unique token is shared among the sites.
- A site is allowed to enter its CS if it possesses the token.
- Token-based algorithms use sequence numbers instead of timestamps. (Used to distinguish between old and current requests.)

Suzuki-Kasami's Broadcast Algorithm

- If a site wants to enter the CS and it does not have the token, it broadcasts a REQUEST message for the token to all other sites.
- A site which possesses the token sends it to the requesting site upon the receipt of its REQUEST message.
- If a site receives a REQUEST message when it is executing the CS, it sends the token only after it has completed the execution of the CS.

continuation..

This algorithm must efficiently address the following two design issues:
(1) How to distinguish an outdated REQUEST message from a current REQUEST message:

- Due to variable message delays, a site may receive a token request message after the corresponding request has been satisfied.
- If a site can not determined if the request corresponding to a token request has been satisfied, it may dispatch the token to a site that does not need it.
- This will not violate the correctness, however, this may seriously degrade the performance.

(2) How to determine which site has an outstanding request for the CS:

- After a site has finished the execution of the CS, it must determine what sites have an outstanding request for the CS so that the token can be dispatched to one of them.

continuation..

The first issue is addressed in the following manner:

- A REQUEST message of site S_j has the form REQUEST(j, n) where n (n=1, 2, ...) is a sequence number which indicates that site S_j is requesting its n^{th} CS execution.
- A site S_i keeps an array of integers RN_i[1..N] where RN_i[j] denotes the largest sequence number received in a REQUEST message so far from site S_j.
- When site S_i receives a REQUEST(j, n) message, it sets RN_i[j]:= max(RN_i[j], n).
- When a site S_i receives a REQUEST(j, n) message, the request is outdated if RN_i[j]>n.

continuation..

The second issue is addressed in the following manner:

- The token consists of a queue of requesting sites, Q, and an array of integers LN[1..N], where LN[j] is the sequence number of the request which site S_j executed most recently.
- After executing its CS, a site S_i updates LN[i]:=RN_i[i] to indicate that its request corresponding to sequence number RN_i[i] has been executed.
- At site S_i if RN_i[j]=LN[j]+1, then site S_j is currently requesting token.

The Algorithm

Requesting the critical section

(a) If requesting site S_i does not have the token, then it increments its sequence number, $RN_i[i]$, and sends a REQUEST(i, *sn*) message to all other sites. ('*sn*' is the updated value of $RN_i[i]$.)

(b) When a site S_j receives this message, it sets $RN_j[i]$ to max($RN_j[i]$, *sn*). If S_j has the idle token, then it sends the token to S_i if $RN_j[i]$=LN[i]+1.

Executing the critical section

(c) Site S_i executes the CS after it has received the token.

The Algorithm

Releasing the critical section Having finished the execution of the CS, site S_i takes the following actions:

(d) It sets LN[i] element of the token array equal to RN_i[i].

(e) For every site S_j whose id is not in the token queue, it appends its id to the token queue if RN_i[j]=LN[j]+1.

(f) If the token queue is nonempty after the above update, S_i deletes the top site id from the token queue and sends the token to the site indicated by the id.

Correctness

Mutual exclusion is guaranteed because there is only one token in the system and a site holds the token during the CS execution.

Theorem: A requesting site enters the CS in finite time.

Proof:

- Token request messages of a site S_i reach other sites in finite time.
- Since one of these sites will have token in finite time, site S_i's request will be placed in the token queue in finite time.
- Since there can be at most $N - 1$ requests in front of this request in the token queue, site S_i will get the token and execute the CS in finite time.

Performance

- No message is needed and the synchronization delay is zero if a site holds the idle token at the time of its request.
- If a site does not hold the token when it makes a request, the algorithm requires N messages to obtain the token. Synchronization delay in this algorithm is 0 or T.