

TO Algorithm – 1

When Basic TO scheduler receives $p_i[x]$:

1. if, for any conflicting operation q , $ts(T_i) < max-q-scheduled[x]$ then:
 - reject $p_i[x]$
 - finish
 2. if, for any conflicting operation q , $0 < q-in-transit[x]$, then:
 - add $p_i[x]$ to $queue[x]$
 - re-sort $queue[x]$ into increasing timestamp order
 - finish
 3. if, for any conflicting operation q , $q_i[x]$ on $queue[x]$, then:
 - add $p_i[x]$ to $queue[x]$
 - re-sort $queue[x]$ into increasing timestamp order
 - finish
 4. otherwise:
 - (a) increment $p-in-transit[x]$
 - (b) set $max-p-scheduled[x]$ to $ts(T_i)$
 - (c) pass $p_i[x]$ to DM for immediate execution
-

TO Algorithm – 2

When a Basic TO scheduler receives an acknowledgement for $u_j[x]$:

1. decrement $u-in-transit[x]$
 2. if $queue[x]$ empty, then:
 - finish
 3. for $p_i[x]$ at start of $queue[x]$:
 - if, for all conflicting operations q , $0 = q-in-transit[x]$, then:
 - (a) remove $p_i[x]$ from $queue[x]$
 - (b) execute step 4 of algorithm on previous slide
 - (c) go to step 2, above
-