Algorithm – Causal-Order Broadcast

Algorithm 1 No-Waiting Causal Broadcast

Implements:
CausalOrderReliableBroadcast, instance crb.

Uses:
ReliableBroadcast, instance rb.

1: upon event ⟨Init⟩ do
2:   delivered := ∅
3:   past := []
4: upon event ⟨crb, Broadcast | m⟩ do
5:   trigger ⟨rb, Broadcast | [DATA, past, m]⟩
6:   past := (self, m) :: past  \> List cons instead of append for brevity.
7: upon event ⟨rb, Deliver | p,[DATA, mpast, m]⟩ do
8:   if m /∈ delivered then
9:     DELIVERDEPS(mpast)
10:    trigger ⟨crb, Deliver | p, m⟩
11:   delivered := delivered ∪ {m}
12:   if (p, m) /∈ past then
13:      past := (p, m) :: past
14: function DELIVERDEPS((p, m) :: rest)
15:   if m /∈ delivered then
16:      DELIVERDEPS(rest)
17:    trigger ⟨crb, Deliver | p, m⟩
18:   delivered := delivered ∪ {m}
19:   if (p, m) /∈ past then
20:      past := (p, m) :: past
Algorithm 2 Broadcast with Sequence Number

Implements:
  
  FIFOReliableBroadcast, instance \textit{frb}.

Uses:
  
  ReliableBroadcast, instance \textit{rb}.

1: \textbf{upon event} (Init) \textbf{do}
2:  lsn := 0
3:  pending := \emptyset
4:  \forall_{p \in \Pi} \text{next}[p] := 1
5: \textbf{upon event} (frb, Broadcast | m) \textbf{do}
6:  lsn := lsn + 1
7:  \textbf{trigger} (rb, Broadcast | [DATA, self, m, lsn])
8: \textbf{upon event} (rb, Deliver | p, [DATA, s, m, sn]) \textbf{do}
9:  pending := pending \cup \{(s, m, sn)\}
10: \textbf{while} \exists_{(s, m', sn') \in \text{pending}} sn' = \text{next}[s] \textbf{do}
11:  next[s] := next[s] + 1
12:  pending := pending \setminus \{(s, m', sn')\}
13:  \textbf{trigger} (frb, Deliver | s, m')
Algorithm 3 No-Waiting Causal Broadcast with FIFO

Implements:
CausalOrderReliableBroadcast, instance crb.

Uses:
FIFO-ReliableBroadcast, instance frb.

1: upon event ( Init ) do
2:     delivered := ∅
3:     l := []
4: upon event ( crb, Broadcast | m ) do
5:     trigger ( frb, Broadcast | [DATA, (self, m) :: l] )
6:     l := []
7: upon event ( frb, Deliver | p, [DATA, l_m] ) do
8:     DELIVERDEPS(l_m)
9: function DELIVERDEPS((p, m) :: rest)
10:     if m ∉ delivered then
11:         DELIVERDEPS(rest)
12:     trigger ( crb, Deliver | p, m )
13:     delivered := delivered ∪ {m}
14:     if (p, m) ∉ l then
15:         l := (p, m) :: l
Algorithm 4 Waiting Causal Broadcast

Implements:
CausalOrderReliableBroadcast, instance \textit{crb}.

Uses:
ReliableBroadcast, instance \textit{rb}.

1: \textbf{upon event }⟨Init⟩\textbf{ do}
2: \hspace{2em} \forall p \in \Pi \hspace{1em} V[p] := 0
3: \hspace{2em} lsn := 0
4: \hspace{2em} pending := \emptyset
5: \textbf{upon event }⟨crb, Broadcast | m⟩\textbf{ do}
6: \hspace{2em} W := V
7: \hspace{2em} W[\text{self}] := lsn
8: \hspace{2em} lsn := lsn + 1
9: \hspace{2em} \textbf{trigger} ⟨rb, Broadcast | [DATA, W, m]⟩
10: \textbf{upon event }⟨rb, Deliver | p, [DATA, W, m]⟩\textbf{ do}
11: \hspace{2em} pending := pending \cup \{(p, W, m)\}
12: \hspace{2em} \textbf{while }\exists (p', W', m') \in \text{pending} \hspace{1em} W' \leq V \hspace{1em} \textbf{do}
13: \hspace{3em} pending := pending \setminus \{(p', W', m')\}
14: \hspace{2em} V[p'] := V[p'] + 1
15: \hspace{2em} \textbf{trigger} ⟨crb, Deliver | p', m'⟩