MODULE *Reno*

Implements simple *AIMD TCP* protocol. It can use any pthe model, provided that it implements the specification of *SimplePathModel*.

EXTENDS *TLC*, *Sequences*, *Integers*

MAX_ARRIVAL : The rate at which packets can arrive. Helps lower runtime.
SSTHRESSH_START: Starting value of Slow-Start Threshold.
MAX_WINDOW : Maximum allowed value of *cwnd*.

added : The number of packets arriving.
arrivals : Total number of packets arrived.
buffered : Number of packets buffered for service.
cwnd : Congestion window.
ssthresh : Slow-Start threshold

CONSTANT *C*, *MAX_T*, *MAX_ARRIVAL*, *DROP_ACK*,
SSTHRESH_START, *MAX_WINDOW*
VARIABLES *t*, *ticked*,
nAck, *inFlight*, *timeout*,
added, *arrivals*, *buffered*, *cwnd*, *ssthresh*

$timeVars \triangleq \langle t, ticked \rangle$
$pathVars \triangleq \langle nAck, inFlight, timeout \rangle$
$tcpVars \triangleq \langle arrivals, buffered, cwnd, ssthresh, added \rangle$
$vars \triangleq \langle t, ticked, nAck, inFlight, timeout, arrivals, added, buffered, cwnd, ssthresh \rangle$

$time \triangleq$ INSTANCE *Time* WITH $t \leftarrow t$, $ticked \leftarrow ticked$, $MAX\_T \leftarrow MAX\_T$

Here, "*SimplePathModel*" is used. Any other path model used must implement:
1) "*ExcessivePacketDropIsEnabled*": A boolean formula that specifies if path capacity has been exceeded. This MUST always trigger a timeout and thus disables the *TCP* until that timeout happens.
2) "*Init*" and "Next" predicates.
3) "Fairness" for fairness conditions if required.
4) Variables $\langle nAck, inFlight, timeout \rangle$ at the very least.

$path \triangleq$ INSTANCE *SimplePathModel* WITH $nAck \leftarrow nAck$, $inFlight \leftarrow inFlight$, $timeout \leftarrow timeout$, $C \leftarrow C$, $MAX\_ARRIVAL \leftarrow MAX\_ARRIVAL$, $DROP\_ACK \leftarrow DROP\_ACK$

ASSUME $\land SSTHRESH\_START > 1$
$\land MAX\_WINDOW > SSTHRESH\_START$
$\land MAX\_T > 2$

A note about "added":

"added" segnifies that some packets have arrived and should go into the input buffer. If added is nonzero, all *TCP* window actions are disabled until we add the packets into the buffer.

This does not change the possible behaviors, but it helps prevent some repeated states when those packets need to be sent.

$$
\begin{aligned}
TypeOK \triangleq\ & \land path\,!\,TypeOK \\
& \land arrivals \in Nat \\
& \land arrivals \geq 0 \\
& \land buffered \in Nat \\
& \land buffered \geq 0 \\
& \land added \in Nat \\
& \land added \geq 0 \\
& \land cwnd \in Nat \\
& \land cwnd \geq 1 \\
& \land ssthresh \in Nat \\
& \land ssthresh \geq 2
\end{aligned}
$$

These are the 3 basic conditions that the specification must satisfy. It gurantees the safety of instantiated specifications.

$$Termination \triangleq path\,!\,Termination$$

$$Finished \triangleq path\,!\,Finished$$

$$RateLimited \triangleq path\,!\,RateLimited$$

$$
\begin{aligned}
Init \triangleq\ & \land path\,!\,Init \\
& \land arrivals = 0 \\
& \land cwnd = 1 \\
& \land ssthresh = SSTHRESH\_START \\
& \land buffered = 0 \\
& \land added = 0
\end{aligned}
$$

EnableIncreaseWindow:

Only enabled when no timeout has happened and we have not exceeded MAX_WINDOW. If still in slow-start, one ACK at least should be present, if during congestion avoidance, we need a whole *cwnd* worth of acks to increase the window.

IncreaseWindow:

The action itself doubles *cwnd* if during slow-start or adds one to it if during congestion avoidance. Appropriate amounts of $ACKs$ are consumed as well ...

$$
\begin{aligned}
EnableIncreaseWindow \triangleq\ & \land timeout = 0 \\
& \land cwnd < MAX\_WINDOW \\
& \land \text{IF } cwnd < ssthresh \\
& \qquad \text{THEN } nAck > 0 \\
& \qquad \text{ELSE } nAck \geq cwnd \\
& \land Finished = \text{FALSE}
\end{aligned}
$$

$$
\begin{aligned}
&\land \neg path!ExcessivePacketDropIsEnabled \\
&\land added = 0
\end{aligned}
$$

$$
\begin{aligned}
IncreaseWindow \triangleq\ & \land EnableIncreaseWindow \\
& \land \text{IF } cwnd < ssthresh \\
& \qquad \text{THEN } \land cwnd' = 2 * cwnd \\
& \qquad\qquad\ \land nAck' = nAck - 1 \\
& \qquad \text{ELSE } \land nAck' = nAck - cwnd \\
& \qquad\qquad\ \land cwnd' = cwnd + 1 \\
& \land \text{UNCHANGED } \langle ssthresh,\ timeout,\ inFlight,\ added, \\
& \qquad\qquad\qquad arrivals,\ buffered,\ t,\ ticked \rangle
\end{aligned}
$$

EnableDecreaseWindow:

Only enabled during timeouts.

DecreaseWindow:

It resets the number of received *ACKs*, sets timeout to zero and cuts the window in half if slow-start is finished, else *cwnd* is set to 1. Half of current *cwnd* is added to slow-start threshold.

$$
\begin{aligned}
EnableDecreaseWindow \triangleq\ & \land timeout = 1 \\
& \land Finished = \text{FALSE} \\
& \land \neg path!ExcessivePacketDropIsEnabled \\
& \land added = 0
\end{aligned}
$$

$$
\begin{aligned}
DecreaseWindow \triangleq\ & \land EnableDecreaseWindow \\
& \land \text{IF } cwnd \geq 4 \\
& \qquad \text{THEN } ssthresh' = cwnd \div 2 \\
& \qquad \text{ELSE } ssthresh' = 2 \\
& \land \text{IF } cwnd < ssthresh \\
& \qquad \text{THEN } \land cwnd' = 1 \\
& \qquad \text{ELSE } \land cwnd' = cwnd \div 2 \\
& \land timeout' = 0 \\
& \land nAck' = 0 \\
& \land \text{UNCHANGED } \langle inFlight,\ added,\ arrivals,\ buffered,\ t, \\
& \qquad\qquad\qquad ticked \rangle
\end{aligned}
$$

$$
Max(a,\ b) \triangleq \text{IF } a > b \text{ THEN } a \text{ ELSE } b
$$

$$
\begin{aligned}
&newPacketsAllowed(timePassed,\ packetsArrived) \triangleq Max(\\
&\quad timePassed * MAX\_ARRIVAL - packetsArrived - 1,\ 0 \\
&)
\end{aligned}
$$

$$
\begin{aligned}
&getRandomArrival(timePassed,\ packetsArrived) \triangleq RandomElement(\\
&\quad 0\ ..\ newPacketsAllowed(timePassed,\ packetsArrived) \\
&)
\end{aligned}
$$

PacketAcceptIsEnabled:

Enabled only when timeout is zero and we have not exceeded the maximum arrival rate yet.

AcceptPackets:

Draws a random number of packets up to a maximum value that gurantees arrival rate is not exceeded, and assigns it to "added". If there are packets to add, all *TCP* actions are disabled until these new packets are added to the input buffer.

$$\begin{aligned} PacketAcceptIsEnabled \triangleq\ & \wedge ticked = 0 \\ & \wedge timeout = 0 \\ & \wedge Finished = \text{FALSE} \\ & \wedge (t * MAX\_ARRIVAL - arrivals) > 0 \\ & \wedge \neg path!ExcessivePacketDropIsEnabled \end{aligned}$$

$$\begin{aligned} AcceptPackets \triangleq\ & \wedge PacketAcceptIsEnabled \\ & \wedge added' = getRandomArrival(t,\ arrivals) \\ & \wedge \text{UNCHANGED } \langle t,\ ticked,\ nAck,\ inFlight,\ timeout,\ cwnd, \\ & \qquad ssthresh,\ arrivals,\ buffered\rangle \end{aligned}$$

PacketArrivalIsEnabled:

Only enabled when added is non-zero and the rest of packet acceptance conditions also hold.

PacketArrival:

Adds the value of "added" to total arrivals and buffered values, and then resets "added" to zero, enabling furthur *TCP* actions.

$$\begin{aligned} PacketArrivalIsEnabled \triangleq\ & \wedge added > 0 \\ & \wedge PacketAcceptIsEnabled \end{aligned}$$

$$\begin{aligned} PacketArrival \triangleq\ & \wedge PacketArrivalIsEnabled \\ & \wedge arrivals' = arrivals + added \\ & \wedge buffered' = buffered + added \\ & \wedge added' = 0 \\ & \wedge \text{UNCHANGED } \langle t,\ ticked,\ nAck,\ inFlight,\ timeout,\ cwnd, \\ & \qquad ssthresh\rangle \end{aligned}$$

PacketSendIsEnabled:

Only enabled when there are buffered packets to deliver and $inFlight < cwnd$, meaning that the window has empty space.

SendNewPackets:

If there are enough buffered packets, the window is filled completely, if not, the window is filled until the buffer is empty.

$$\begin{aligned} PacketSendIsEnabled \triangleq\ & \wedge ticked = 0 \\ & \wedge Finished = \text{FALSE} \\ & \wedge buffered\ > 0 \\ & \wedge inFlight\ < cwnd \end{aligned}$$

$$
\begin{aligned}
&\land\ timeout = 0 \\
&\land\ \neg path!ExcessivePacketDropIsEnabled \\
&\land\ added = 0
\end{aligned}
$$

$$
\begin{aligned}
SendNewPackets \triangleq\ & \land\ PacketSendIsEnabled \\
& \land \text{IF}\ cwnd - inFlight > buffered \\
& \qquad \text{THEN}\ \land\ inFlight' = inFlight + buffered \\
& \qquad\qquad\ \land\ buffered' = 0 \\
& \qquad \text{ELSE}\ \land\ buffered' = buffered - (cwnd - inFlight) \\
& \qquad\qquad\ \land\ inFlight' = cwnd \\
& \land \text{UNCHANGED}\ \langle t,\ ticked,\ nAck,\ timeout,\ cwnd,\ added, \\
& \qquad\qquad ssthresh,\ arrivals \rangle
\end{aligned}
$$

$$
\begin{aligned}
Next \triangleq\ & \lor\ IncreaseWindow \\
& \lor\ DecreaseWindow \\
& \lor\ SendNewPackets \\
& \lor\ AcceptPackets \\
& \lor\ PacketArrival \\
& \lor\ \land\ path!Next \\
& \quad\ \land \text{UNCHANGED}\ tcpVars
\end{aligned}
$$

$$
Fairness \triangleq\ \land\ path!Fairness
$$

$$
Spec \triangleq\ Init \land \Box[Next]_{vars} \land Fairness
$$

\ * Modification History
\ * Last modified *Thu Nov* 03 18:58:18 *IRST* 2022 by *Arvin*
\ * Created *Mon Oct* 31 01:51:50 *IRST* 2022 by *Arvin*