─────────────── MODULE *scheduler* ───────────────

EXTENDS *TLC*, *Sequences*, *SequencesExt*, *FiniteSets*

CONSTANTS *Subscribers*, *Servers*, *Clients*, *Workers*

$AllTask \triangleq Subscribers \cup Subscribers$

   **--algorithm** *scheduler*
**variables**
     events
    $wait\_set = \{\}$;

     states of tasks
    $run\_queue = \langle \rangle$;
    $running = \{\}$;
    $waiting\ = AllTask$;

**define**
    $starvation\_free \triangleq \forall\, event \in AllTask : event \in wait\_set \rightsquigarrow \diamond(event \in running)$
**end define**

**fair + process** *scheduler* = "scheduler"
**begin**
    *start_sched*:
      **while** TRUE **do**
          **await** $wait\_set \neq \{\}$;

          pick runnable tasks and change the states to *run_queue* from waiting
          **with** $tasks = waiting \cap wait\_set,$
              $servers = tasks \cap Servers,$
              $clients\ = tasks \cap Clients,$
              $subscribers = tasks \cap Subscribers$ **do**
              push to *run_queue*
              $run\_queue := run\_queue \circ SetToSeq(servers) \circ SetToSeq(clients) \circ SetToSeq(subs$

              change state
              $waiting := ((waiting \setminus servers) \setminus clients) \setminus subscribers$;
          **end with** ;
      **end while** ;
**end process** ;

**fair process** *trigger_subscriber* $\in$ *Subscribers*
**begin**
    *start_subscriber*:
      **while** TRUE **do**
        $wait\_set := wait\_set \cup \{self\}$;
      **end while** ;
**end process** ;

1

**fair process** *trigger_server* $\in$ *Servers*
**begin**
    *start_server*:
        **while** TRUE **do**
            *wait_set* := *wait_set* $\cup$ {*self*} **;**
        **end while** **;**
**end process** **;**

**fair process** *trigger_client* $\in$ *Clients*
**begin**
    *start_client*:
        **while** TRUE **do**
            *wait_set* := *wait_set* $\cup$ {*self*} **;**
        **end while** **;**
**end process** **;**

 worker thread
**fair** + **process** *worker* $\in$ *Workers*
**variables**
    *task* **;**
**begin**
     work-stealing
    *start_worker*:
        **while** TRUE **do**
            **await** *run_queue* $\neq$ $\langle\rangle$   **;**

            *task* := *Head*(*run_queue*) **;**
            *run_queue* := *Tail*(*run_queue*) **;**
            *running* := *running* $\cup$ {*task*} **;**

            *finish_task*:
                *running* := *running* $\setminus$ {*task*} **;**
                *waiting* := *waiting* $\cup$ {*task*} **;**
        **end while** **;**
**end process** **;**

**end algorithm** **;**
 BEGIN TRANSLATION (*chksum(pcal)* = "3af7a303" $\wedge$ *chksum(tla)* = "a6607055")
CONSTANT *defaultInitValue*
VARIABLES *wait_set*, *run_queue*, *running*, *waiting*, *pc*

 define statement
*starvation_free* $\triangleq$ $\forall$ *event* $\in$ *AllTask* : *event* $\in$ *wait_set* $\rightsquigarrow$ $\diamond$(*event* $\in$ *running*)

VARIABLE *task*

*vars* $\triangleq$ $\langle$*wait_set*, *run_queue*, *running*, *waiting*, *pc*, *task*$\rangle$

$ProcSet \triangleq \{\text{"scheduler"}\} \cup (Subscribers) \cup (Servers) \cup (Clients) \cup (Workers)$

$Init \triangleq$   Global variables
$\quad\quad \wedge wait\_set = \{\}$
$\quad\quad \wedge run\_queue = \langle \rangle$
$\quad\quad \wedge running = \{\}$
$\quad\quad \wedge waiting = AllTask$
   Process worker
$\quad\quad \wedge task = [self \in Workers \mapsto defaultInitValue]$
$\quad\quad \wedge pc = [self \in ProcSet \mapsto \text{CASE } self = \text{"scheduler"} \rightarrow \text{"start\_sched"}$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \square \quad self \in Subscribers \rightarrow \text{"start\_subscriber"}$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \square \quad self \in Servers \rightarrow \text{"start\_server"}$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \square \quad self \in Clients \rightarrow \text{"start\_client"}$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \square \quad self \in Workers \rightarrow \text{"start\_worker"}]$

$start\_sched \triangleq \wedge pc[\text{"scheduler"}] = \text{"start\_sched"}$
$\quad\quad\quad\quad \wedge wait\_set \neq \{\}$
$\quad\quad\quad\quad \wedge \text{LET } tasks \triangleq waiting \cap wait\_set \text{ IN}$
$\quad\quad\quad\quad\quad \text{LET } servers \triangleq tasks \cap Servers \text{ IN}$
$\quad\quad\quad\quad\quad\quad \text{LET } clients \triangleq tasks \cap Clients \text{ IN}$
$\quad\quad\quad\quad\quad\quad\quad \text{LET } subscribers \triangleq tasks \cap Subscribers \text{ IN}$
$\quad\quad\quad\quad\quad\quad\quad\quad \wedge run\_queue' = run\_queue \circ SetToSeq(servers) \circ SetToSeq(clients) \circ SetToSeq(s$
$\quad\quad\quad\quad\quad\quad\quad\quad \wedge waiting' = ((waiting \setminus servers) \setminus clients) \setminus subscribers$
$\quad\quad\quad\quad \wedge pc' = [pc \text{ EXCEPT } ![\text{"scheduler"}] = \text{"start\_sched"}]$
$\quad\quad\quad\quad \wedge \text{UNCHANGED } \langle wait\_set, running, task \rangle$

$scheduler \triangleq start\_sched$

$start\_subscriber(self) \triangleq \wedge pc[self] = \text{"start\_subscriber"}$
$\quad\quad\quad\quad\quad\quad\quad \wedge wait\_set' = (wait\_set \cup \{self\})$
$\quad\quad\quad\quad\quad\quad\quad \wedge pc' = [pc \text{ EXCEPT } ![self] = \text{"start\_subscriber"}]$
$\quad\quad\quad\quad\quad\quad\quad \wedge \text{UNCHANGED } \langle run\_queue, running, waiting, task \rangle$

$trigger\_subscriber(self) \triangleq start\_subscriber(self)$

$start\_server(self) \triangleq \wedge pc[self] = \text{"start\_server"}$
$\quad\quad\quad\quad\quad\quad \wedge wait\_set' = (wait\_set \cup \{self\})$
$\quad\quad\quad\quad\quad\quad \wedge pc' = [pc \text{ EXCEPT } ![self] = \text{"start\_server"}]$
$\quad\quad\quad\quad\quad\quad \wedge \text{UNCHANGED } \langle run\_queue, running, waiting, task \rangle$

$trigger\_server(self) \triangleq start\_server(self)$

$start\_client(self) \triangleq \wedge pc[self] = \text{"start\_client"}$
$\quad\quad\quad\quad\quad\quad \wedge wait\_set' = (wait\_set \cup \{self\})$
$\quad\quad\quad\quad\quad\quad \wedge pc' = [pc \text{ EXCEPT } ![self] = \text{"start\_client"}]$
$\quad\quad\quad\quad\quad\quad \wedge \text{UNCHANGED } \langle run\_queue, running, waiting, task \rangle$

$trigger\_client(self) \triangleq start\_client(self)$

$start\_worker(self)\ \triangleq\ \wedge\ pc[self] =$ "start_worker"
$\qquad\qquad\qquad\qquad \wedge\ run\_queue \neq \langle\rangle$
$\qquad\qquad\qquad\qquad \wedge\ task' = [task\ \text{EXCEPT}\ ![self] = Head(run\_queue)]$
$\qquad\qquad\qquad\qquad \wedge\ run\_queue' = Tail(run\_queue)$
$\qquad\qquad\qquad\qquad \wedge\ running' = (running \cup \{task'[self]\})$
$\qquad\qquad\qquad\qquad \wedge\ pc' = [pc\ \text{EXCEPT}\ ![self] =$ "finish_task"$]$
$\qquad\qquad\qquad\qquad \wedge\ \text{UNCHANGED}\ \langle wait\_set,\ waiting \rangle$

$finish\_task(self)\ \triangleq\ \wedge\ pc[self]\ \ =$ "finish_task"
$\qquad\qquad\qquad\qquad \wedge\ running' = running \setminus \{task[self]\}$
$\qquad\qquad\qquad\qquad \wedge\ waiting'\ = (waiting \cup \{task[self]\})$
$\qquad\qquad\qquad\qquad \wedge\ pc' = [pc\ \text{EXCEPT}\ ![self] =$ "start_worker"$]$
$\qquad\qquad\qquad\qquad \wedge\ \text{UNCHANGED}\ \langle wait\_set,\ run\_queue,\ task \rangle$

$worker(self)\ \triangleq\ start\_worker(self) \vee finish\_task(self)$

$Next\ \triangleq\ scheduler$
$\qquad\qquad \vee\ (\exists\, self \in Subscribers : trigger\_subscriber(self))$
$\qquad\qquad \vee\ (\exists\, self \in Servers\ \ : trigger\_server(self))$
$\qquad\qquad \vee\ (\exists\, self \in Clients\ \ \ : trigger\_client(self))$
$\qquad\qquad \vee\ (\exists\, self \in Workers : worker(self))$

$Spec\ \triangleq\ \wedge\ Init \wedge \Box[Next]_{vars}$
$\qquad\qquad \wedge\ \text{SF}_{vars}(scheduler)$
$\qquad\qquad \wedge\ \forall\, self \in Subscribers : \text{WF}_{vars}(trigger\_subscriber(self))$
$\qquad\qquad \wedge\ \forall\, self \in Servers\ \ : \text{WF}_{vars}(trigger\_server(self))$
$\qquad\qquad \wedge\ \forall\, self \in Clients\ \ \ : \text{WF}_{vars}(trigger\_client(self))$
$\qquad\qquad \wedge\ \forall\, self \in Workers : \text{SF}_{vars}(worker(self))$

END TRANSLATION