
MODULE *UpdateCluster*

EXTENDS *Integers, FiniteSets*

CONSTANTS

_Requests, the requests sent by the user
_Workers, the pool of workers
NULL

VARIABLES

lastVOK, last successfully applied version
toApply, the version to apply (last requests that passed the initial tests)
cluster, cluster state
requests, the state of all requests
workers, the state of all workers
lock damn, I used a lock...

VARIABLES

these variables are *lla* + details
confOK, are we able to get a valid conf ?
reqCounter just to keep track of the order of submissions

vars \triangleq $\langle \text{confOK}, \text{reqCounter}, \text{lastVOK}, \text{toApply}, \text{cluster}, \text{requests}, \text{workers}, \text{lock} \rangle$

TypeInvariants \triangleq

$\wedge \text{confOK} \in \text{BOOLEAN}$ won't change for a specific behavior
 $\wedge \text{lock} \in \text{BOOLEAN}$
 $\wedge \text{cluster.st} \in \{$
 "idle",
 "starting",
 "partial",
 "failed"
 $\}$
 $\wedge \forall r \in \text{_Requests} : \text{requests}[r].\text{st} \in \{$
 "waiting", the request (req) hasn't been submitted yet
 "submitted", req has been submitted
 "rejected", req has been rejected (auth problem)
 "valid" auth etc passed
 $\}$
 $\wedge \forall w \in \text{_Workers} : \text{workers}[w].\text{st} \in \{$
 "waiting",
 "starting",
 "working"
 $\}$

Initial State

$Init \triangleq$
 $\wedge requests = [r \in _Requests \mapsto [st \mapsto \text{"waiting"}, v \mapsto NULL]]$
 $\wedge workers = [w \in _Workers \mapsto [st \mapsto \text{"waiting"}, v \mapsto NULL]]$
 $\wedge cluster = [v \mapsto 0, st \mapsto \text{"idle"}]$
 $\wedge lastVOK = 0$
 $\wedge reqCounter = 0$
 $\wedge toApply = 0$
 $\wedge confOK \in \text{BOOLEAN}$
 $\wedge lock = \text{FALSE}$

Actions

$Submit(r) \triangleq$ update request received from the user
 $LET\ newV \triangleq reqCounter + 1\ IN$
 $\wedge requests[r].st = \text{"waiting"}$
 $\wedge reqCounter' = newV$
 $\wedge requests' = [requests\ EXCEPT\ ![r].st = \text{"submitted"}, ![r].v = newV]$
 $\wedge UNCHANGED\ \langle confOK, lastVOK, toApply, cluster, workers, lock \rangle$

$Initialcheck(r) \triangleq$ request validation (auth, quotas...)
 $\wedge requests[r].st = \text{"submitted"}$
 $\wedge \exists ok \in \text{BOOLEAN} :$
 $\quad IF\ ok$
 $\quad \quad THEN$
 $\quad \quad \quad requests' = [requests\ EXCEPT\ ![r].st = \text{"valid"}]$
 $\quad \quad ELSE$
 $\quad \quad \quad requests' = [requests\ EXCEPT\ ![r].st = \text{"rejected"}]$
 $\wedge UNCHANGED\ \langle confOK, reqCounter, lastVOK, toApply, cluster, workers, lock \rangle$

$PushToPending(r) \triangleq$ the request is pushed to queue
 $\wedge requests[r].st = \text{"valid"}$
 $\wedge IF\ toApply < requests[r].v$
 $\quad THEN\ \wedge toApply' = requests[r].v$
 $\quad \quad \wedge UNCHANGED\ \langle confOK, reqCounter, lastVOK, cluster, requests, workers, lock \rangle$
 $\quad ELSE\ \wedge requests' = [requests\ EXCEPT\ ![r].st = \text{"rejected"}]$
 $\quad \quad \wedge UNCHANGED\ \langle confOK, reqCounter, lastVOK, toApply, cluster, workers, lock \rangle$

$SpawnWorker(w) \triangleq$ spawns a new worker
 $\wedge workers[w].st = \text{"waiting"}$
 $\wedge toApply \neq lastVOK$
 $\wedge lock = \text{FALSE}$
 $\wedge \vee cluster.st = \text{"idle"}$

$\vee cluster.st = \text{"failed"}$
 $\wedge workers' = [workers \text{ EXCEPT } ![w].v = toApply, ![w].st = \text{"starting"}]$
 $\wedge lock' = \text{TRUE}$
 $\wedge \text{UNCHANGED } \langle confOK, reqCounter, lastVOK, toApply, requests, cluster \rangle$

$ApplyStart(w) \triangleq$ the cluster starts to be modified
 $\wedge workers[w].st = \text{"starting"}$
 $\wedge \text{IF } workers[w].v \geq toApply$
 THEN
 IF $confOK$
 THEN
 $\wedge cluster' = [v \mapsto workers[w].v, st \mapsto \text{"partial"}]$
 $\wedge workers' = [workers \text{ EXCEPT } ![w].st = \text{"working"}]$
 $\wedge \text{UNCHANGED } \langle confOK, reqCounter, lastVOK, toApply, requests, lock \rangle$
 ELSE
 $\wedge lock' = \text{FALSE}$
 $\wedge workers' = [workers \text{ EXCEPT } ![w].st = \text{"waiting"}, ![w].v = NULL]$
 $\wedge \text{UNCHANGED } \langle confOK, reqCounter, lastVOK, toApply, cluster, requests \rangle$
 ELSE a new version has been submitted, no need to apply this one
 $\wedge workers' = [workers \text{ EXCEPT } ![w].st = \text{"waiting"}, ![w].v = NULL]$
 $\wedge lock' = \text{FALSE}$
 $\wedge \text{UNCHANGED } \langle confOK, reqCounter, lastVOK, toApply, cluster, requests \rangle$

$RollbackVersion \triangleq$
 to differentiate it from the original last VOK
 $lastVOK + 10$

$ApplyFinish(w) \triangleq$ the cluster update finishes
 $\wedge workers[w].st = \text{"working"}$
 $\wedge lock' = \text{FALSE}$
 $\wedge \exists ok \in \text{BOOLEAN} :$
 IF $ok \vee workers[w].v = RollbackVersion$ rollback always works
 THEN
 $\wedge cluster' = [cluster \text{ EXCEPT } !.st = \text{"idle"}]$
 $\wedge lastVOK' = workers[w].v$
 $\wedge workers' = [workers \text{ EXCEPT } ![w].st = \text{"waiting"}, ![w].v = NULL]$
 $\wedge \text{UNCHANGED } \langle confOK, reqCounter, toApply, requests \rangle$
 ELSE
 $\wedge cluster' = [cluster \text{ EXCEPT } !.st = \text{"failed"}]$
 $\wedge workers' = [workers \text{ EXCEPT } ![w].st = \text{"waiting"}, ![w].v = NULL]$
 $\wedge \text{IF } workers[w].v < toApply$
 THEN a newer version has been submitted
 $\wedge \text{UNCHANGED } \langle confOK, reqCounter, lastVOK, toApply, requests \rangle$
 ELSE let's trigger a rollback

$$\begin{aligned} &\wedge toApply' = RollbackVersion \\ &\wedge \text{UNCHANGED } \langle confOK, reqCounter, lastVOK, requests \rangle \end{aligned}$$

Requirements

$$NoConcurrentUpdate \triangleq$$

$$\square(\text{Cardinality}(\{r \in \text{DOMAIN } requests : requests[r].st = \text{"working"}\}) < 2)$$

$$NoPartialUpdateTermination \triangleq$$

$$\diamond\square(cluster.st = \text{"idle"})$$

we don't want the cluster to end up in a partially update *st*

$$EveryReqIsProcessed \triangleq$$

$$\diamond\square(\neg\exists r \in _Requests : requests[r].st = \text{"waiting"})$$

Spec

$$Next \triangleq$$

$$\begin{aligned} &\vee \exists r \in _Requests : \\ &\quad \vee Submit(r) \\ &\quad \vee Initialcheck(r) \\ &\quad \vee PushToPending(r) \\ &\vee \exists w \in _Workers : \\ &\quad \vee SpawnWorker(w) \\ &\quad \vee ApplyStart(w) \\ &\quad \vee ApplyFinish(w) \end{aligned}$$

$$Fairness \triangleq \forall r \in _Requests, w \in _Workers :$$

$$\begin{aligned} &\wedge WF_{vars}(Submit(r)) \\ &\wedge WF_{vars}(Initialcheck(r)) \\ &\wedge WF_{vars}(PushToPending(r)) \\ &\wedge WF_{vars}(SpawnWorker(w)) \\ &\wedge WF_{vars}(ApplyStart(w)) \\ &\wedge WF_{vars}(ApplyFinish(w)) \end{aligned}$$

$$Spec \triangleq$$

$$\begin{aligned} &\wedge Init \\ &\wedge \square[Next]_{vars} \\ &\wedge Fairness \end{aligned}$$

THEOREM $Spec \Rightarrow \square(\text{TypeInvariants})$

THEOREM $Spec \Rightarrow NoPartialUpdateTermination$

THEOREM $Spec \Rightarrow NoApplicationOfOutdatedReq$

THEOREM $Spec \Rightarrow EveryReqInQueueIsProcessed$