compObs(currentBsState)

```
1: $currentContext \leftarrow context[currentBsState]$
2: for each state in states[currentContext] do
3:    $tran \leftarrow trOut[state]$
4:    while $tran \neq NIL$ do
5:       if $isEventPresent(currentContext, tran)$ and $isBufferFree(currentContext, tran)$ and
         $isTransitionObservable(currentContext, tran)$ then
6:          $newContext \leftarrow createNewContext(currentContex, tran)$
7:          $newTransition \leftarrow createNewTransition(currentContext, tran)$
8:          $item \leftarrow contextSearch(newContext, ctHashMap)$
9:          if $item \neq NIL$ then
10:            $destinationBsState \leftarrow subValue[item]$
11:            $dest[newTransition] \leftarrow destinationBsState$
12:         else
13:            $createNewState(newContext, tran)$
14:            $step(destinationBsState)$
15:         end if
16:      end if
17:   end while
18: end for
```

takeEventFromBuffer(context,action)

```
1: $l \leftarrow link[action]$
2: $pos \leftarrow index[l]$
3: return $buffer[context][pos]$
```

isEventPresent(context,transition)

```
1: $actionRequest \leftarrow actIn[transition]$
2: $eventBuffer \leftarrow takeEventFromBuffer(context, actionRequest)$
3: $eventRequest \leftarrow event[actionRequest]$
4: return $(actionRequest = NIL$ or $eventBuffer = eventRequest)$
```

isBufferFree(context,transition)

```
1: $actionProduced \leftarrow actOut(transition)$
2: while $actionProduced \neq NIL$ do
3:   $eventBuffer \leftarrow takeEventFromBuffer(context, actionProduced)$
4:   if $eventBuffer \neq NIL$ then
5:     return $FALSE$
6:   end if
7:   $actionProduced \leftarrow next[actionProduced]$
8: end while
9: return $TRUE$
```

createNewContext(context,transition)

```
1: $newContext \leftarrow initializeContext()$
2: $state \leftarrow dest[transition]$
3: $actionRequest \leftarrow actIn[transition]$
4: $eventRequest \leftarrow event[actionRequest]$
5: if $eventRequest \neq NIL$ then
6:   $eventBuffer \leftarrow NIL$
7: end if
8: $actionProduced \leftarrow actOut[tran]$
9: while $actionProduced \neq NIL$ do
10:   $l2 \leftarrow link[actionProduced]$
11:   $pos2 \leftarrow index[l2]$
12:   $buffer[newContext][pos2] \leftarrow actionProduced$
13:   $actionProduced \leftarrow next[actionProduced]$
14: end while
15: * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
16: $obsIndex[newContext] \leftarrow obsIndex[newContext] + 1$
17: $currentObs[newContext] \leftarrow prev[currentObs[newContext]]$
18: {la lista parte dalla coda e poi va verso la testa, gli eventi sono specchiati rispetto ai valori di input}
19: * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * **
20: return $newContext$
```

createNewTransition(context,transition)

```
1: $newTransition \leftarrow initializeTransition()$
2: $obs[newTransition] \leftarrow obs[transition]$
3: $rel[newTransition] \leftarrow rel[transition]$
4: $src[newTransition] \leftarrow createSource(context)$
5: $netBs \leftarrow addTransition(newTransition)$
6: return newTransition
```

createNewState(context,transition)

```
1: $destinationBsState \leftarrow initializeState()$
2: $context[destinationBsState] \leftarrow context$
3: if $isFinal(context)$ then
4:    $final[destinationBsState] \leftarrow TRUE$
5: else
6:    $finale[destinationBsState] \leftarrow FALSE$
7: end if
8: ************************************
9: if $currentObs[context] \neq NIL$ then
10:    $final[destinationBsState] \leftarrow FALSE$
11: end if
12: ************************************
13: $netBs \leftarrow addState(destinationBsState)$
14: $dest[transition] \leftarrow destinationBsState$
15: $addContextToHashMap(context)$
```

dfsVisit(state)

```
1: $color[source] \leftarrow GRAY$
2: $transitionsIncoming \leftarrow trIn[state]$
3: while $transitionsIncoming \neq NIL$ do
4:    $stateSource \leftarrow scr[transitionsIncoming]$
5:    if $color[stateSource] = WHITE$ then
6:       dfsVisit[stateSource]
7:    end if
8:    $transitionsIncoming \leftarrow nect[transitionsIncoming]$
9: end while
10: $color[state] \leftarrow BLACK$
```

prune(network)

```
1: {consideriamo solo reti con un solo automa, infatti il pruning avviene solo sulle
2: network generate dal sistema}
3: $autom \leftarrow automatons[network]$
4: $totalStat \leftarrow states[autom]$
5: {La lista autom avrà un solo elemento}
6: while $totalStat \neq NIL$ do
7:    if $final[totalState]$ then
8:       $dfsVisit(totalState)$
9:    end if
10:    $totalState \leftarrow next[totalState]$
11: end while
12: while $totalStat \neq NIL$ do
13:    if $color[totalState] = WHITE$ then
14:       $removeTheState(network, totalState)$
15:    end if
16:    $totalState \leftarrow next[totalState]$
17: end while
```

isTransitionObservable(context,transition)

```
label $\leftarrow NIL$
currentObservation $\leftarrow$ currentObs[context]
{Controllo sulla presenza della lista di osservazioni}
**if** currentObservation $\neq NIL$ **then**
    label $\leftarrow$ currentObervation
    transitionLabel $\leftarrow$ obs[tansition]
    idObsarvation $\leftarrow$ id[transitionLabel]
**end if**
**if** transitionLabel $\neq NIL$ **and** (label $\neq NIL$ **or** idLabel $\neq$ idObsarvation) **then**
    **return** FALSE
**else**
    **return** TRUE
**end if**
```