State - Search Conditions

```
(deftemplate status
        (slot water-jug
                (type INTEGER)
                (default 0)
                (range 0 5))
        (slot closed
                (type INTEGER)
                (default 0)
                (range 0 1))
        (slot heuristic
                (type INTEGER)
                (default 0))
        (slot depth
                (type INTEGER)
                (default 0))
        (slot node
                (type INTEGER)
                (default 0))
        (slot parent
                (type FACT-ADDRESS SYMBOL)
                (default no-parent)
                (allowed-symbols no-parent))
        (slot operation
                (type STRING))
        )
```

The Heuristic

```
;-------------------------------------------------------------
; The Heuristic Function
;
(deffunction calculate-heuristic (?amount ?node ?depth)
        (if (> ?amount 3) then
                -1
        else
                (+ 1 ?amount)))
```

Initialization

```
;-----------------------------------------------------------
;  The initial Status
;        The heuristic is set low
;        Depth is zero
;        Node is zero
;        Water Jug has no water in it
;
(defrule Initial-State
        ?initial <- (initial-fact)
=>
        (assert (heuristic 1))
        (assert (depth 0 0))
        (assert (status (water-jug 0)
                        (depth 0)
                        (node 0)
                        (heuristic 0)
                        (parent no-parent)))
        (assert (nodes 1))
        )
```

Initialization

```
CLIPS> (reset)
CLIPS> (run 1)
CLIPS> (facts)
f-0     (initial-fact) CF 1.00
f-1     (heuristic 1) CF 1.00
f-2     (depth 0 0) CF 1.00
f-3     (status (water-jug 0)
               (closed 0) (heuristic 0) (depth 0)
               (node 0) (parent no-parent) (operation "")) CF 1.00
f-4     (nodes 1) CF 1.00
For a total of 5 facts.
CLIPS> (agenda)
700    Increment-Node-Count: f-3,f-4
700    update-heuristic-value: f-3
600    top-heuristic: f-1,f-3
150    increment-heuristic: f-1
100    update-depth: f-2,f-4
For a total of 5 activations.
```

Node Count Increment

```
;-----------------------------------------------------------
; A new fact is created and it is assigned a node number
;  The node number is then incremented
;
(defrule Increment-Node-Count
        (declare (salience 700))
        ?status <- (status (node ?statusnode))
        ?nodes <- (nodes ?nodecount)
        (test (= ?statusnode 0))
=>
        (assert (nodes (+ 1 ?nodecount)))
        (modify ?status (node ?nodecount))
        (retract ?nodes)
        )
```

Update Heuristic In Node

```
;   If a node with a higher heuristic is created -
;        Update heuristic value
(defrule update-heuristic-value
        (declare (salience 700))
        ?node <- (status (heuristic 0)
                         (node ?num)
                         (depth ?depth)
                         (water-jug ?amount))
=>
        (modify ?node (heuristic (calculate-heuristic ?amount ?num ?dept

;---------------------------------------------------------
; The Heuristic Function
;
(deffunction calculate-heuristic (?amount ?node ?depth)
        (if (> ?amount 3) then
                -1
        else
                (- 4 ?amount)))
```

Heuristic in Node

```
CLIPS> (facts)
f-0     (initial-fact) CF 1.00
f-2     (depth 0 0) CF 1.00
f-5     (nodes 2) CF 1.00
f-7     (status (water-jug 0) (closed 0) (heuristic 4)
                (depth 0) (node 1) (parent no-parent) (operation "")) CF
f-8     (heuristic 2) CF 1.00
```

```
;-------------------------------------------------------------
;  We are not done, but all nodes  of current heuristic value
;     have been expanded:
;           Find a heuristic value where facts match
(defrule increment-heuristic
        (declare (salience 150))
        ?current <- (heuristic ?currentvalue)
=>
        (retract ?current)
        (assert (heuristic (+ ?currentvalue 1))))

f-0     (initial-fact) CF 1.00
f-2     (depth 0 0) CF 1.00
f-5     (nodes 2) CF 1.00
f-7     (status (water-jug 0) (closed 0) (heuristic 4)
                (depth 0) (node 1) (parent no-parent) (operation "")) CF
f-10    (heuristic 4) CF 1.00
```

Get Rule to Expand

```
CLIPS> (agenda)
400    get-best-rule-to-expand: f-7,f-10
150    increment-heuristic: f-10
100    update-depth: f-2,f-5
For a total of 3 activations.
```

Next Rule: The closed field

```
;---------------------------------------------------------
; Start Expansion of a Fact to produce new nodes
;---------------------------------------------------------
; This moves the rule which has the same value as the
;  heuristic to the CLOSED list (i.e. closed set to 1)
;  The initial firing activates in usage in the rules
;
(defrule get-best-rule-to-expand
        (declare (salience 400))
        ?status <- (status (closed ?closed)
                           (heuristic ?value))
        (heuristic ?heuristic)
        (test (= 0 ?closed))
        (test (= ?heuristic ?value))
=>
        (modify ?status (closed 1)))


500    Add-One-Gallon: f-11
500    Add-Two-Gallons: f-11
150    increment-heuristic: f-10
100    update-depth: f-2,f-5
```

Rule to Expand Node

```
;-------------------------------------------------------------
; Rule for Adding One Gallon to Water Jug
;
(defrule Add-One-Gallon
        (declare (salience 500))
        ?status <- (status
                        (closed ?closed)
                        (water-jug ?amount)
                        (depth ?depth))
        (test (= ?closed 1))
=>
        (assert (status (water-jug (+ ?amount 1))
                        (depth (+ ?depth 1))
                        (parent ?status)
                        (operation "Add One Gallon")
                        ))
        )
```

```
700    Increment-Node-Count: f-12,f-5
700    update-heuristic-value: f-12
600    top-heuristic: f-10,f-12
500    Add-Two-Gallons: f-11
150    increment-heuristic: f-10
100    update-depth: f-2,f-5

f-0     (initial-fact) CF 1.00
f-2     (depth 0 0) CF 1.00
f-10    (heuristic 4) CF 1.00
f-11    (status (water-jug 0) (closed 1) (heuristic 4)
               (depth 0) (node 1) (parent no-parent) (operation "")) CF
f-13    (nodes 3) CF 1.00
f-15    (status (water-jug 1) (closed 0) (heuristic 3)
                (depth 1) (node 2) (parent <Fact-11>) (operation "Add On
For a total of 6 facts.
```

Update Heuristic

```
;-------------------------------------------------------------
;  If a node was created that was better than the
;     current heuristic, then update it to this value
(defrule top-heuristic
        (declare (salience 600))
        ?current <- (heuristic ?currentvalue)
        ?node    <- (status (heuristic ?nodevalue) (closed ?closed))
        (test (= 0 ?closed))
        (test (< ?nodevalue ?currentvalue))
=>
        (retract ?current)
        (assert (heuristic ?nodevalue)))
```

```
CLIPS> (agenda)
500    Add-Two-Gallons: f-11
400    get-best-rule-to-expand: f-15,f-16
150    increment-heuristic: f-16
100    update-depth: f-2,f-13
For a total of 4 activations.
CLIPS> (run 1)
CLIPS> (run 1)
CLIPS> (run 1)
CLIPS> (run 1)
CLIPS> (facts)
f-0     (initial-fact) CF 1.00
f-2     (depth 0 0) CF 1.00
f-11    (status (water-jug 0) (closed 1) (heuristic 4)
                (depth 0) (node 1) (parent no-parent) (operation "")) CF
f-15    (status (water-jug 1) (closed 0) (heuristic 3)
                (depth 1) (node 2) (parent <Fact-11>) (operation "Add On
f-18    (nodes 4) CF 1.00
f-20    (status (water-jug 2) (closed 0) (heuristic 2) (depth 1) (node 3
f-21    (heuristic 2) CF 1.00
For a total of 7 facts.



400    get-best-rule-to-expand: f-20,f-21
150    increment-heuristic: f-21
100    update-depth: f-2,f-18
For a total of 3 activations.
```

Pick Rule with Best Heuristic: f-22

```
f-0     (initial-fact) CF 1.00
f-2     (depth 0 0) CF 1.00
f-11    (status (water-jug 0) (closed 1) (heuristic 4) (depth 0) (node 1
f-15    (status (water-jug 1) (closed 0) (heuristic 3) (depth 1) (node 2
f-18    (nodes 4) CF 1.00
f-21    (heuristic 2) CF 1.00
f-22    (status (water-jug 2) (closed 1) (heuristic 2) (depth 1) (node 3

500    Add-One-Gallon: f-22
500    Add-Two-Gallons: f-22
150    increment-heuristic: f-21
100    update-depth: f-2,f-18
```

Impossible Rule Created

```
f-31    (status (water-jug 4) (closed 0)
              (heuristic -1) (depth 2) (node 5)
              (parent <Fact-22>) (operation "Add Two Gallons")) CF 1.00

;-------------------------------------------------------------
;  If the heuristic of the node says it will not lead to a
;      solution, then retract it
(defrule eliminate-illegal
        (declare (salience 600))
        ?node <- (status (heuristic ?value))
        (test (< ?value 0))
=>
        (retract ?node))
```

Goal Found

```
f-26    (status (water-jug 3) (closed 0) (heuristic 1)
                (depth 2) (node 4) (parent <Fact-22>)
                (operation "Add One Gallon")) CF 1.00

;-------------------------------------------------------------
; Rule for Detecting Done with Task (i.e. the jug has 3 gallons in it)
; Put Salience High so when matched, it is the first rule
;
(defrule goal-found
        (declare (salience 450))
        ?status <- (status
                        (heuristic ?heuristic)
                        (water-jug 3))
        (test (> ?heuristic 0))
=>
        (modify ?status (heuristic -1000)))

;-------------------------------------------------------------
; When a goal is found, its heuristic is set to -1000
;   this starts the printout loop
(defrule Done
        (declare (salience 2000))
        ?status <- (status (heuristic ?heuristic))
        (test (= ?heuristic -1000))
=>
        (assert (complete ?status))
        (assert (list "Done")))
```

```
After 5 nodes; Solution:
("Add Two Gallons" "Add One Gallon")


CLIPS> (run)
After 7 nodes; Solution:
("Add One Gallon" "Add Two Gallons")
After 9 nodes; Solution:
("Add One Gallon" "Add One Gallon" "Add One Gallon")
A Total of 9 Nodes in the Tree
```

Finished

```
;-------------------------------------------------------------
;  Finished is defined when the cost exceeds a given number
(defrule finished
        (declare (salience 160))
        (heuristic ?heuristic)
        (test (> ?heuristic 10))
        (nodes ?nodes)
=>
        (printout t "A Total of " (- ?nodes 1) " Nodes in the Tree" crlf
        (halt)
        )
```