Modeling, Specification, and Simulation of QLOCK in CafeOBJ

FUTATSUGI, Kokichi
二木 厚吉
JAIST

Topics

- What are Mutual Exclusion Protocol and QLOCK?
- Modeling and Description of QLOCK
- Formal specification of QLOCK in CafeOBJ
- Simulation of QLOCK with built-in search Predicate

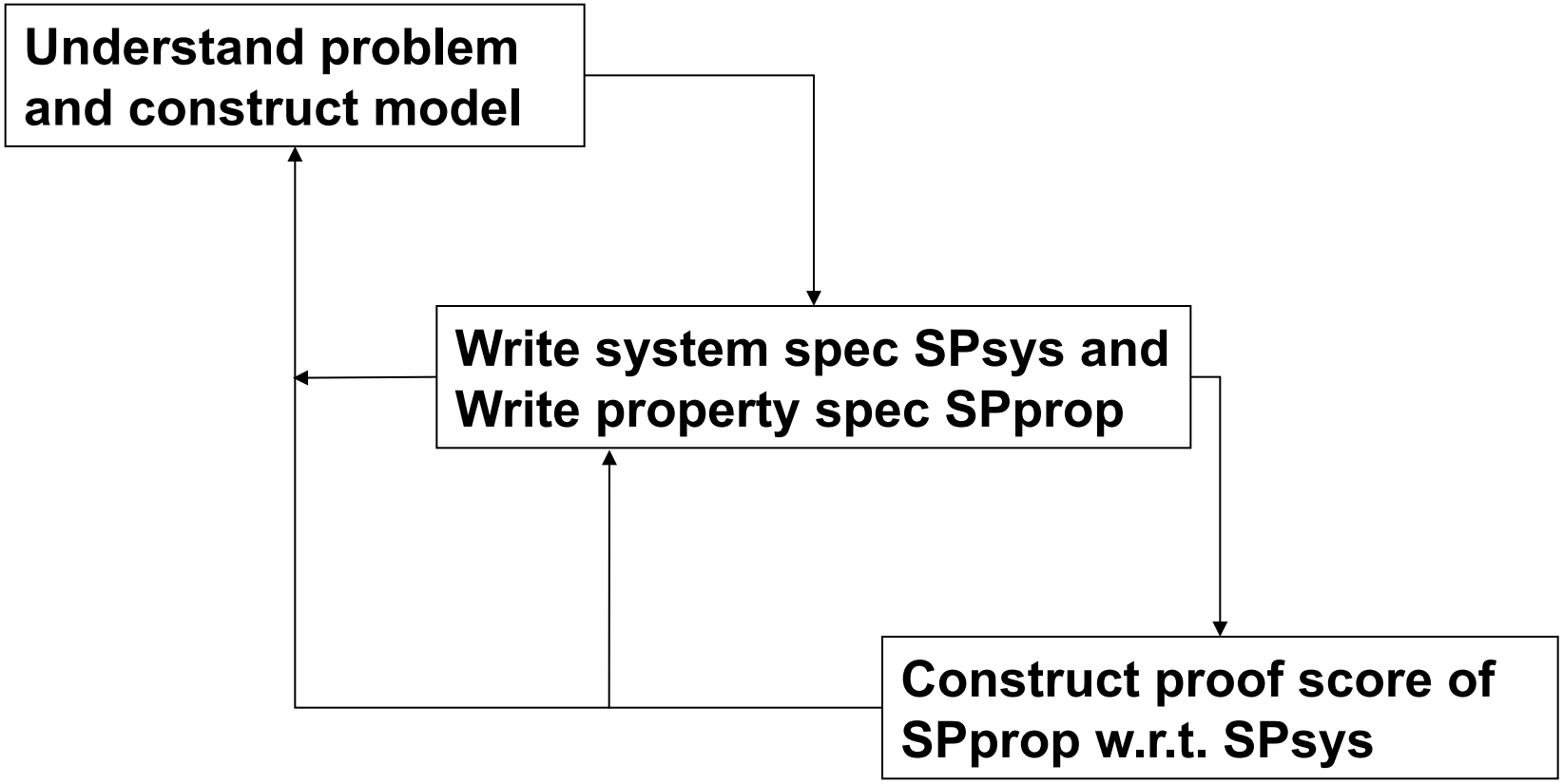
Modeling, Specifying, and Verifying (MSV) in CafeOBJ

1. By understanding a problem to be modeled/ specified, determine several sorts of objects (entities, data, agents, states) and operations (functions, actions, events) over them for describing the problem

2. Define the meanings/functions of the operations by declaring equations over expressions/terms composed of the operations

3. Write proof scores for properties to be verified

MSV with proof scores in CafeOBJ



Mutual Exclusion Protocol

Assume that many agents (or processes) are competing for a common equipment, but at any moment of time only one agent can use the equipment. That is, the agents are mutually excluded in using the equipment. A protocol (mechanism or algorithm) which can achieve the mutual exclusion is called "mutual exclusion protocol".

QLOCK (Mutual Exclusion Protocol by Locking with Queue)

Each of unbounded number of agents who participates in the protocol behaves as follows:

- If the agent wants to use the common equipment and its name is not in the queue yet, put its name at the bottom of the queue.
- If the agent wants to use the common equipment and its name is already in the queue, check if its name is on the top of the queue. If its name is on the top of the queue, start to use the common equipment. If its name is not on the top of the queue, wait until its name is on the top of the queue.
- If the agent finishes to use the common equipment, remove its name from the top of the queue.

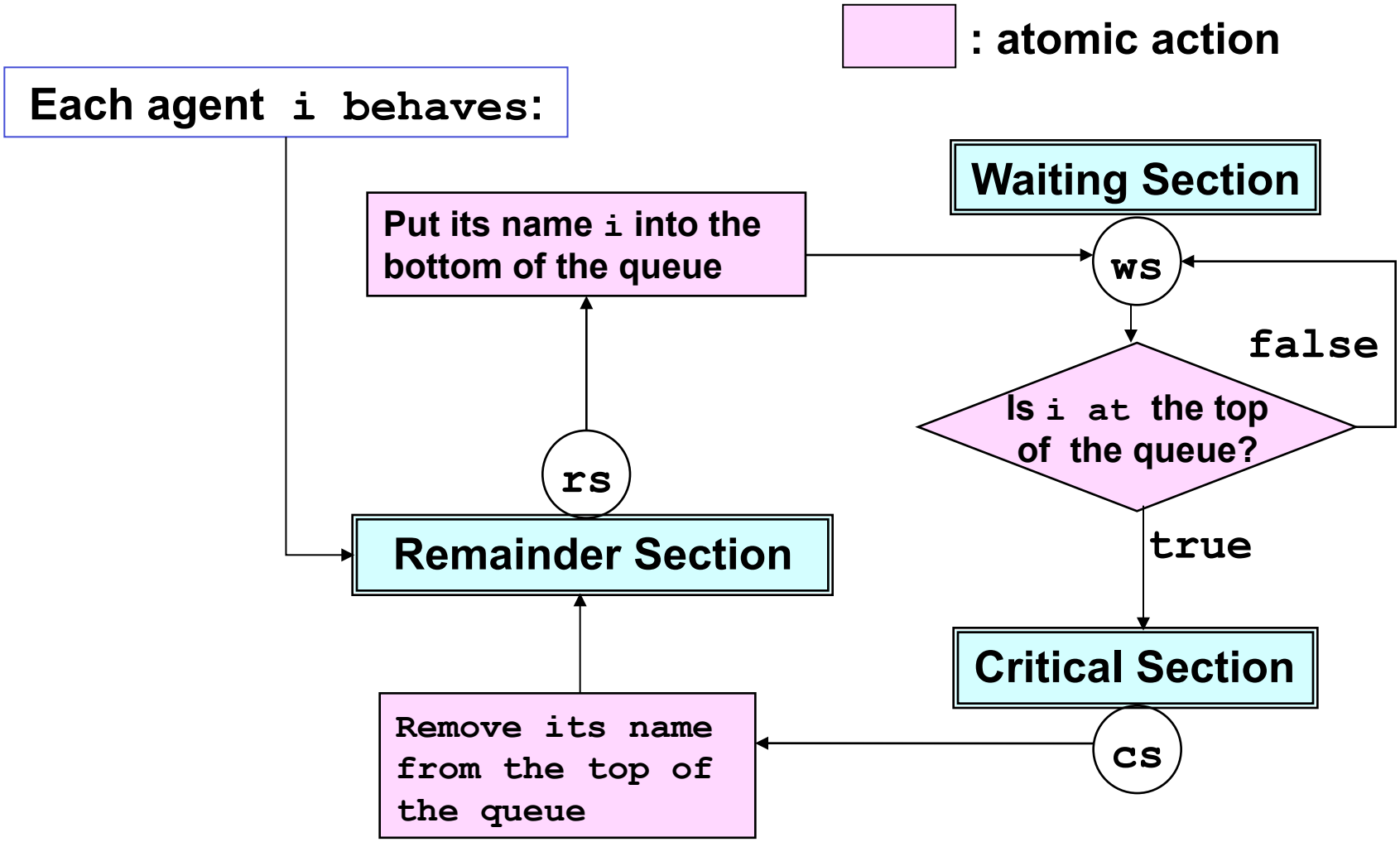
The protocol starts from the state with the empty queue.

QLOCK: basic assumptions/characteristics

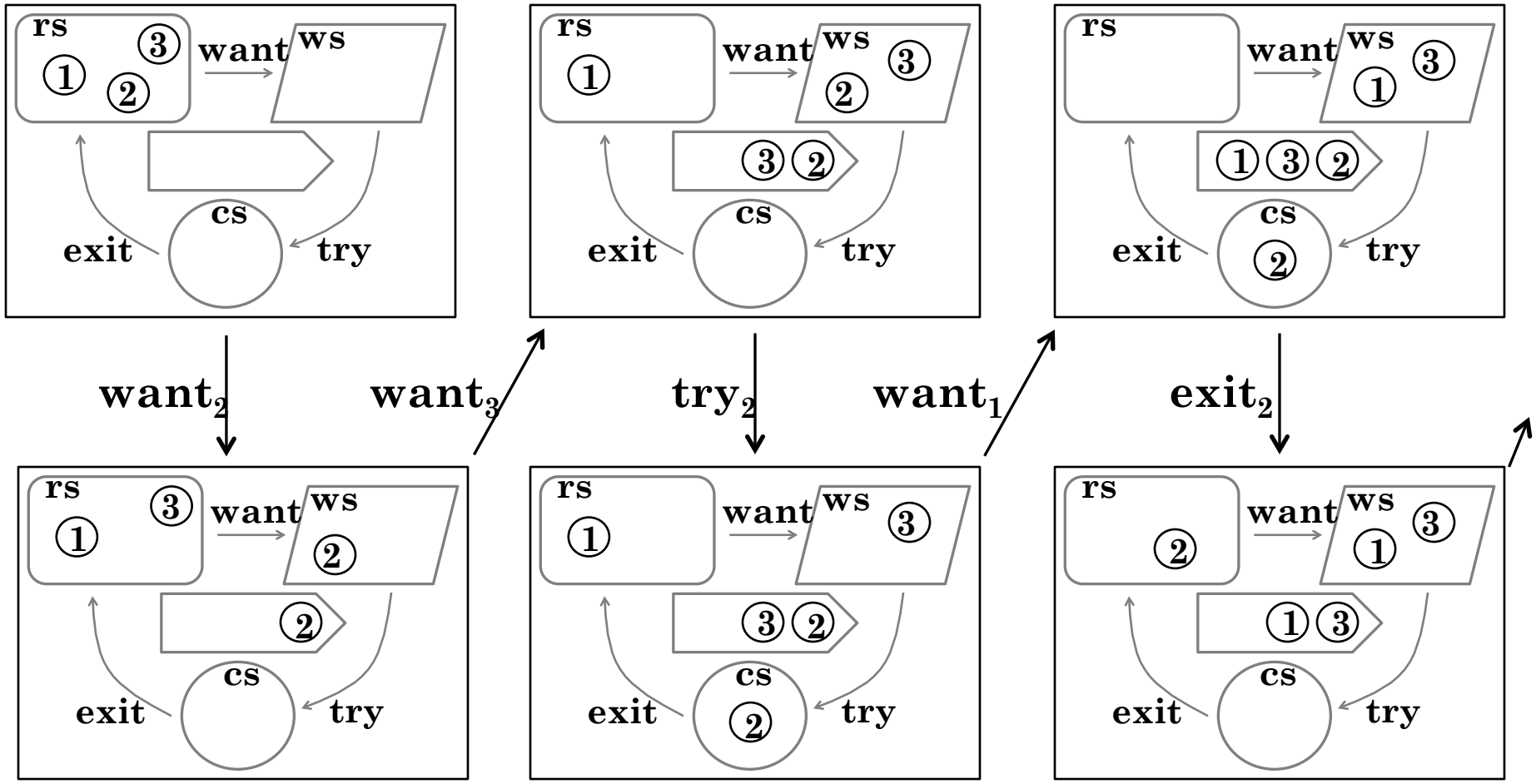
- There is only one queue and all agents/processes share the queue.
- Any basic action on the queue is inseparable (or atomic). That is, when any action is executed on the queue, no other action can be executed until the current action is finished.
- There may be unbounded number of agents.
- In the initial state the queue is empty.

The property to be shown is that at most one agent is using the common equipment at any time.

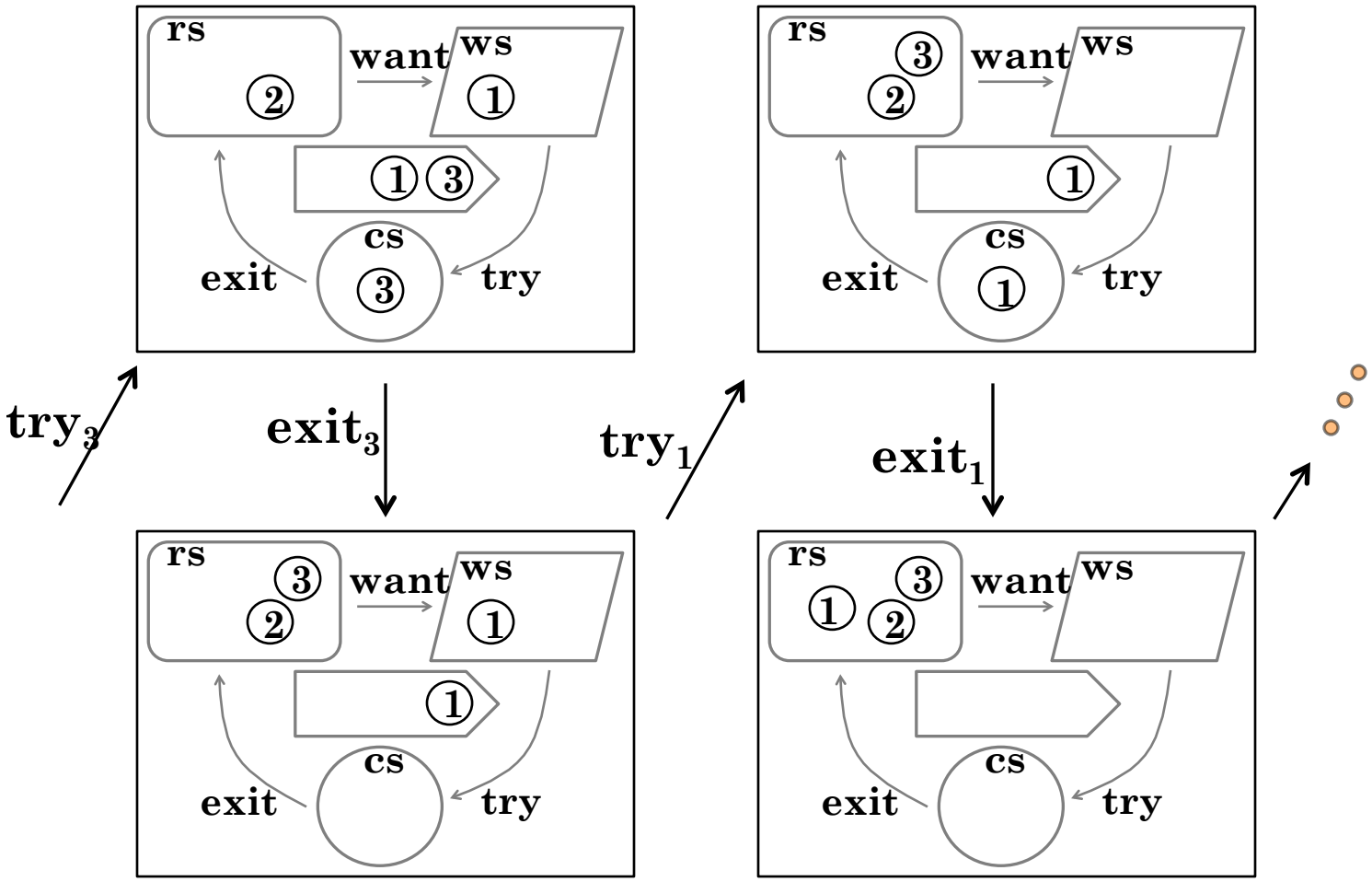
QLOCK (locking with queue): a mutual exclusion protocol



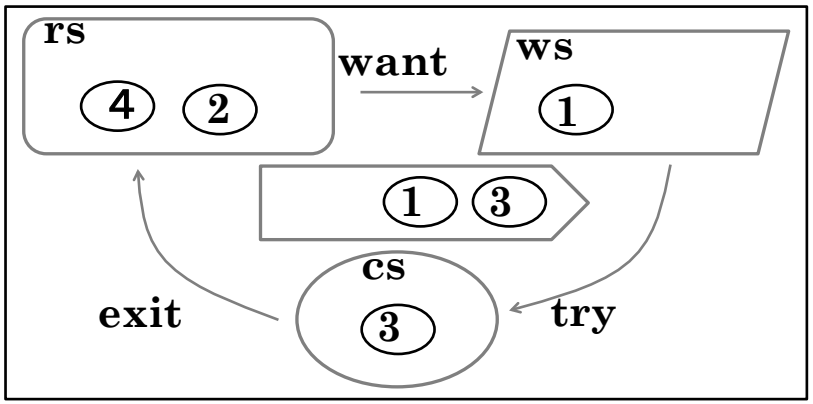
Animation of QLOCK (1)



Animation of QLOCK (2)



State Configuration for QLOCK



≀

[(③ | ①) r ② ④ w ① c ③]

```
[State]
op [_r_w_c_] : Aq As As As -> State {constr}
```

(Aq: AgentQueue As: AgentSet)

QLOCK Transitions in CafeOBJ

```
-- wt: want transition
mod! WT {pr(STATE)
tr[wt]:
         [AQ:Aq      r (A1:Aid AS1:As) w AS2:As    c AS3:As]
      => [(AQ | A1) r AS1              w (A1 AS2) c AS3] . }
-- ty: try transition
mod! TY {pr(STATE)
tr[ty]:
         [(A:Aid | AQ:Aq) r AS1:As w (A AS2:As) c AS3:As]
      => [(A | AQ)         r AS1    w AS2        c (A AS3)] .}
-- exc: exit transition by conditional transition rule
mod! EXC {pr(STATE)
ctr[exc]:
         [(A:Aid | AQ:Aq) r AS1:As   w AS2:As c (A3:Aid AS3:As)]
      => [AQ              r (A3 AS1) w AS2    c AS3]
         if (A = A3) . }
```

QLOCK System Specification in CafeOBJ

```
seq.cafe
set.cafe
qlock-sys.cafe
```

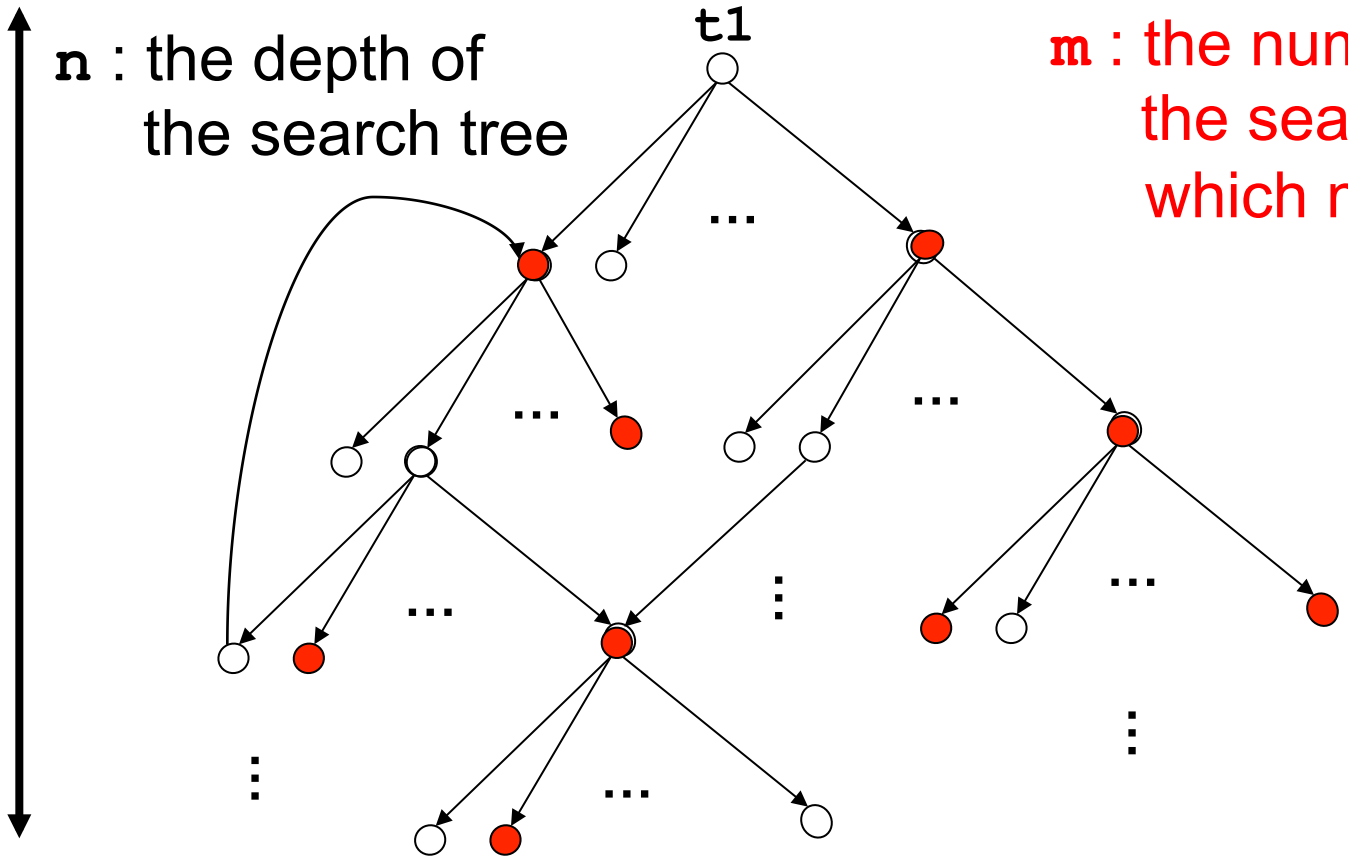
Built-in Search Predicate of CafeOBJ

CafeOBJ System has the following built-in predicate:
- **Any** is any sort (that is, the predicate is available for any sort)
- **NzNat*** is a built-in sort containing non-zero natural number and the special symbol "*" which stands for infinity

```
pred _=(_,_)=>*_ : Any NzNat* NzNat* Any
```

(t1 =(m,n)=>* t2) returns **true** if **t1** can be translated (or rewritten), via more than 0 times transitions **with trans rules**, to some term which matches to **t2** and all matched terms are printed out. Otherwise, it returns **false**. Possible transitions are searched in breadth first fashion. **n** is upper bound of the depth of the search, and **m** is upper bound of the number of terms which match to **t2**. If either of the depth of the search or the number of the matched terms reaches to the upper bound, the search stops.

t1 =(m,n)=>* t2


n : the depth of
the search tree
t1
m : the number of
the searched terms
which match to t2

Search All Reachable States

```
--> search all reachable states with one agent
red [empQ r b1 w empS c empS] =(*,*)=>*
    S:State .

--> search all reachable states with two agents
red [empQ r (b1 b2) w empS c empS] =(*,*)=>*
    S:State .

--> search all reachable states with three agents
red [empQ r (b1 b2 b3) w empS c empS] =(*,*)=>*
    S:State .
```

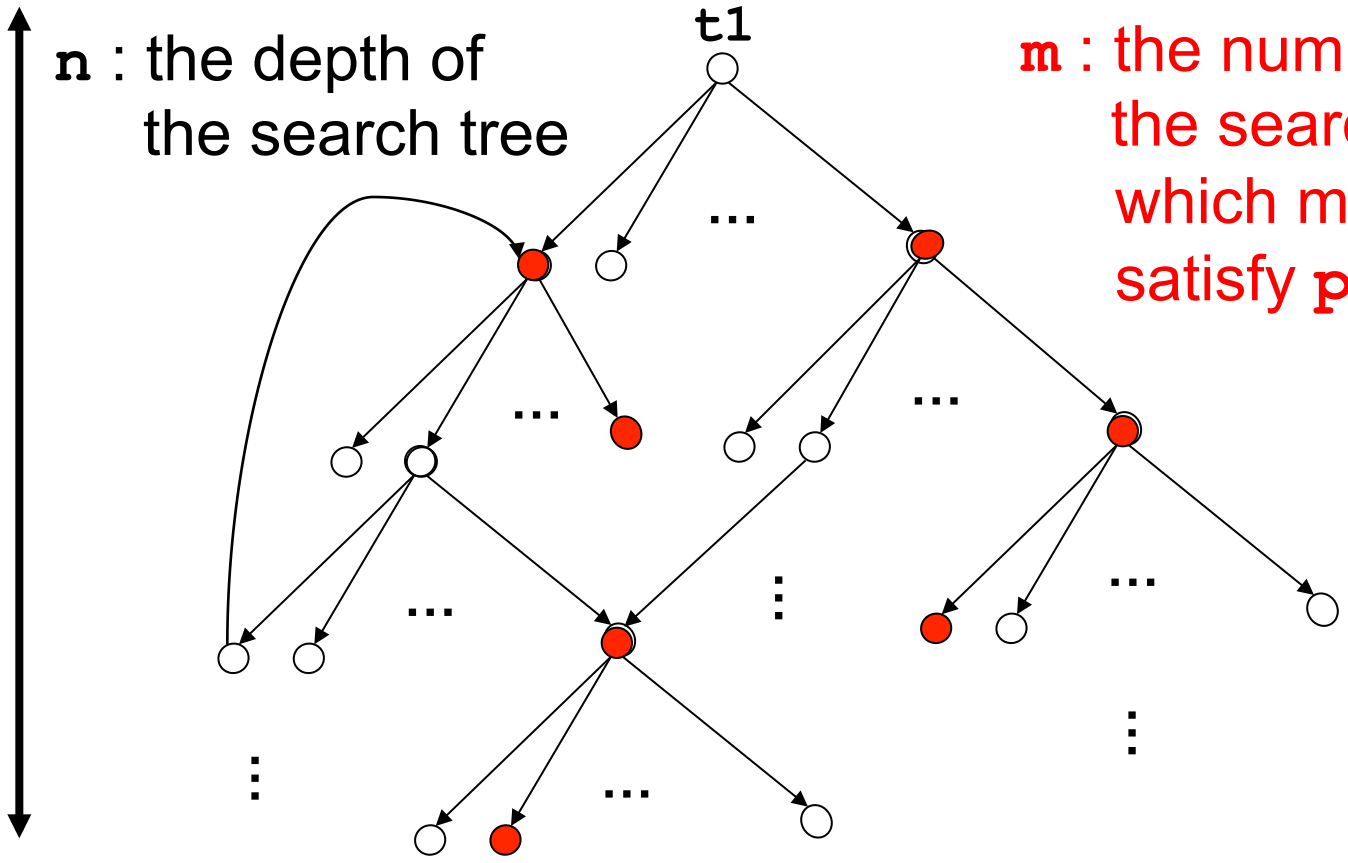
Search Predicate with suchThat

```
t1 =(m,n)=>* t2 suchThat pred1(t2)
```

`pred1(t2)` is a predicate about **`t2`** and can refer to the variables which appear in **`t2`**.

`pred1(t2)` enhances the condition used to determine the term which matches to **t2**.

t1 =(m,n)=>* t2 suchThat pred1(t2)



Check All Reachable States with 3 Agents for Mutual Exclusion Property

```
--> check whether mutual exclusion property holds for
--> all reachable states with 3 agents
red not([empQ r (b1 b2 b3) w empS c empS]
         =(*,*)=>*
        [AQ:Aq r ASr:As w ASw:As c ASc:As]
        suchThat (not(mx ASc))) .
```

If it returns ‘true’ then the mutual exclusion property is verified for QLOCK with 3 agents.