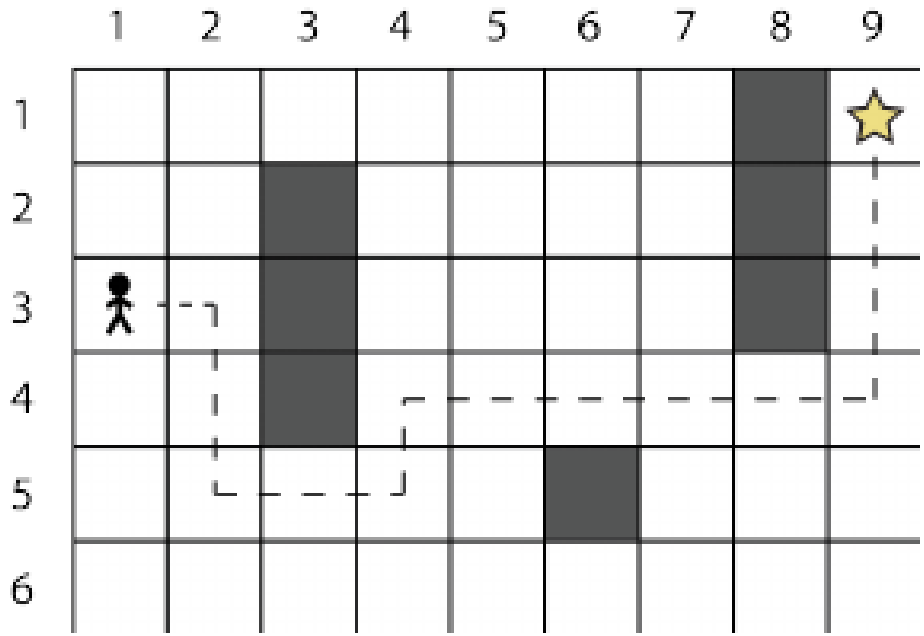
Basic grid-world

Consider an environment represented by a **NxN grid**. An agent can **move in any of the 8 adjacent cells unless they are occupied by an obstacle**. The **cost of each move is one**. The agent **starts** from the **lower leftmost cell** and should **reach** the **upper rightmost cell**.

Formalize the problem of finding a sequence of agent moves to reach the goal.



Basic grid-world

Domain

```
(define (domain basic-grid)
  (:requirements :strips)
  (:predicates (at ?x) (adj ?x ?y) (obs ?x))

  (:action move-agent
    :parameters (?from ?to)
    :precondition (and (at ?from)
                       (adj ?from ?to)
                       (not (obs ?to)))
    :effect (and (not (at ?from)) (at ?to)))

)
```

Basic grid-world

Problem Generic

```
(define (problem basic-grid)
  (:domain basic-grid)
  (:objects          sq-1-1 sq-1-2 … sq-1-N
                              sq-2-1 sq-2-2 … sq-2-N
                                        …
                              sq-N-1 sq-N-2 … sq-N-N)

  (:init     (adj sq-1-1 sq-1-2) (adj sq-1-2 sq-1-1)
                    (adj sq-1-1 sq-2-1) (adj sq-2-1 sq-1-1)
                    (adj sq-1-1 sq-2-2) (adj sq-2-2 sq-1-1) … )

             (obs sq-1-3) (obs sq-2-2) ….

             (at sq-N-1)
  )
  (:goal (and (at sq-1-N)))
)
```

Basic grid-world

Problem 2x3 implementation

```
(define (problem basic_grid-problem)
 (:domain basic_grid-domain)
 (:objects sq-1-1 sq-1-2  sq-1-3
        sq-2-1 sq-2-2  sq-2-3)
 (:init
    ;;row1 adj
    (adj sq-1-1 sq-1-2) (adj sq-1-2 sq-1-3)
    (adj sq-1-2 sq-1-1) (adj sq-1-3 sq-1-2)
    ;;diagonals adj
    (adj sq-1-1 sq-2-2) (adj sq-2-2 sq-1-1)
    (adj sq-2-1 sq-1-2) (adj sq-1-2 sq-2-1)
    (adj sq-1-2 sq-2-3) (adj sq-2-3 sq-1-2)
    (adj sq-2-2 sq-1-3) (adj sq-1-3 sq-2-2)
    ;;cols adj
    (adj sq-1-1 sq-2-1) (adj sq-1-2 sq-2-2) (adj sq-1-3 sq-2-3)
    (adj sq-2-1 sq-1-1) (adj sq-2-2 sq-1-2) (adj sq-2-3 sq-1-3)
    ;;row2 adj
    (adj sq-2-1 sq-2-2) (adj sq-2-2 sq-2-3)
    (adj sq-2-2 sq-2-1) (adj sq-2-3 sq-2-2)

    ;;obs and agent
    (obs sq-1-1) (obs sq-1-2)
    (at sq-2-1)
 )
 (:goal (and (at sq-1-3)))
)
```