A 1.



Given the state connection to be, as shown above, we can now run DFS on the same.

Initial State = S1

Pop from stack = S1 ……… 1 Goal Check

Add successors of S1 -> S0, S2

Pop from stack = S0 ……… 1 Goal Check

No successors of S0

Pop from stack = S2 ……… 1 Goal Check

Add successors of S2 -> S0, S3

Pop from stack = S0 ……… 1 Goal Check

No successors of S0

Pop from stack = S3 ……… 1 Goal Check

Add successors of S3 -> S0, S4

…..

…..

…..

…..

Pop from stack = S0 ……… 1 Goal Check

No successors of S0

Pop from stack = Sn-1 ……… 1 Goal Check

Add successors of Sn-1 -> S0, Sn

Pop from stack = S0 ……… 1 Goal Check

No successors of S0

Pop from stack = Sn ……… 1 Goal Check

Goal Found

Thus, the goal checks are made for

S1, S0, S2, S0, S3, …. S0, Sn-1, S0, Sn

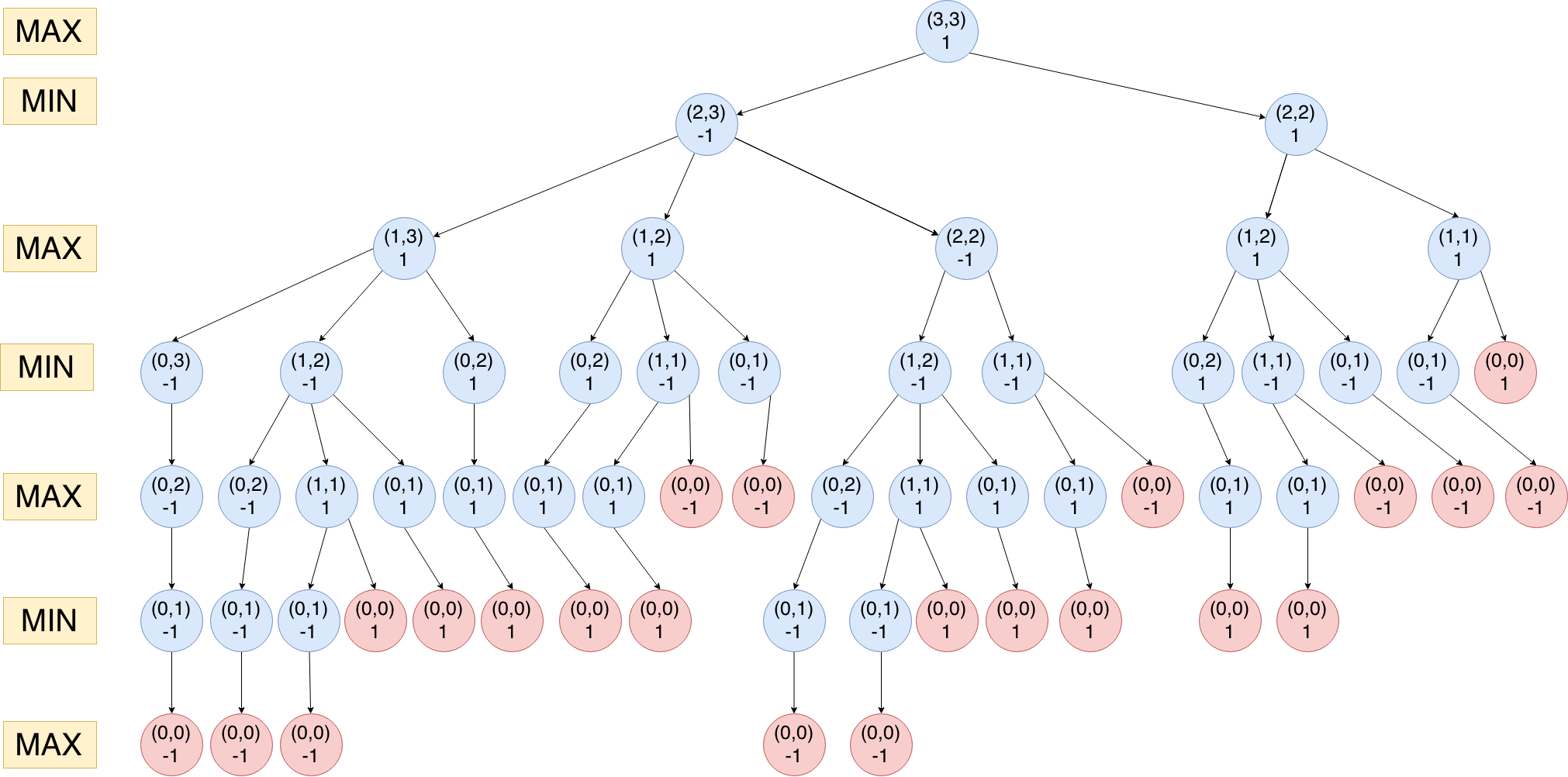
Total no of goal checks = 1 + 2 + 2 + ….. (n-2+1)times … + 2

Total no of goal checks = 1 + 2(n-1)

Total no of goal checks = 1 + 2n - 2

**Total no of goal checks = 2n - 1**

A 2.



A 3.

P(H|A) = a

P(H|B) = b

P(A) = 0.5

P(B) = 0.5

Event (E) = H, T

To calculate: P(A|E) = P(E|A) / P(E)

now, P(E) = P(E,A) + P(E,B)

=> P(E) = P(E|A).P(A) + P(E|B).P(B)

=> P(E) = a.(1-a).0.5 + b.(1-b).0.5 {given the toin cosses are independent}

=> P(A|E) =

**=> P(A|E) =**

A 4.

u1 =

u2 =

Projected data point = (1, 2)

Let us assume, the original point is (x1, x2)

=> The projected data point = (u1Tx, u2Tx)

=> u1Tx = 1, and u2Tx = 2

=> …….. (1)

,and ……. (2)

adding (1) and (2)

=>

=>

=>

subtracting (2) from (1)

=>

=>

=>

**hence, the original point is**

A 6.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | a(0,0) | b(1,0) | c(3,1) | d(7,7) | e(9,9) | f(3,6) |
| a(0,0) |  | 1 | 3.162278 | 9.899495 | 12.727922 | 6.708204 |
| b(1,0) |  |  | 2.236068 | 9.219544 | 12.041595 | 6.324555 |
| c(3,1) |  |  |  | 7.211103 | 10 | 5 |
| d(7,7) |  |  |  |  | 2.828427 | 4.123106 |
| e(9,9) |  |  |  |  |  | 6.708204 |
| f(3,6) |  |  |  |  |  |  |

Cluster 1: (a+b)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | a(0,0)+b(1,0) | c(3,1) | d(7,7) | e(9,9) | f(3,6) |
| a(0,0)+b(1,0) |  | 2.236068 | 9.219544 | 12.041595 | 6.324555 |
| c(3,1) |  |  | 7.211103 | 10 | 5 |
| d(7,7) |  |  |  | 2.828427 | 4.123106 |
| e(9,9) |  |  |  |  | 6.708204 |
| f(3,6) |  |  |  |  |  |

Cluster 1: (a+b)

Cluster 2: (c + (a+b))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | a(0,0)+b(1,0)+c(3,1) | d(7,7) | e(9,9) | f(3,6) |
| a(0,0)+b(1,0)+c(3,1) |  | 7.211103 | 10 | 5 |
| d(7,7) |  |  | 2.828427 | 4.123106 |
| e(9,9) |  |  |  | 6.708204 |
| f(3,6) |  |  |  |  |

Cluster 1: (a+b)

Cluster 2: (c + (a+b))

Cluster 3: (d+e)

|  |  |  |  |
| --- | --- | --- | --- |
|  | a(0,0)+b(1,0)+c(3,1) | d(7,7)+e(9,9) | f(3,6) |
| a(0,0)+b(1,0)+c(3,1) |  | 7.211103 | 5 |
| d(7,7)+e(9,9) |  |  | 4.123106 |
| f(3,6) |  |  |  |

Cluster 1: (a+b)

Cluster 2: (c + (a+b))

Cluster 3: (d+e)

Cluster 4: (f + (d+e))

|  |  |  |
| --- | --- | --- |
|  | a(0,0)+b(1,0)+c(3,1) | d(7,7)+f(3,6)+e(9,9) |
| a(0,0)+b(1,0)+c(3,1) |  | 5 |
| d(7,7)+f(3,6)+e(9,9) |  |  |

Cluster 1: (a+b)

Cluster 2: (c + (a+b))

Cluster 3: (d+f)

Cluster 4: (e + (d+f))

Cluster 5: ( (c + (a+b)) + (e + (d+f)) )

A 7.

Let x = (x1, x2) ∈ R2

objective function f(x) =  
stepsize η = 0.1

= (1, 1)

= ?

From gradient descent, we know

= - η∇f()

∇f(x) = ∇()

= (

=

now, ∇f() =

=

=

=

= (1,1) - 0.1\*∇f()

= (1,1) - 0.1\*

= (1,1) -

= (0.8 - 0.1\*,1 - 0.1\*)

**= (0.8 - 0.1\*,1 - 0.1\*)**

A 8.

σ(x) =

σ’(x) =

Using the quotient and chain rule, we can differentiate above.

Hence,

σ’(x) =

σ’(x) =

σ’(x) =

σ’(x) =

σ’(x) =

σ’(x) =

σ’(x) =

σ’(x) =

A 9.

The optimal policy for our case is

π(s1) = right

π(s2) = right

π(s3) = right

...

...

...

…

π(sn-1) = right

π(sn) = left

Now,

v(s1) = 𝞬0.R0 + 𝞬1.R1 + 𝞬2.R2 + 𝞬3.R3 + ……

v(s1) = 𝞬0.0 + 𝞬1.0 + 𝞬2.0 + 𝞬3.0 + …… 𝞬n-3.0 + 𝞬n-2.1 + 𝞬n-1.0 + 𝞬n.1 + 𝞬n+1.0 + 𝞬n+2.1 + ….

v(s1) = 𝞬n-2.1 + 𝞬n.1 + 𝞬n+2.1 + ….

**v(s1) =**

A 10.

Given that: q(s, a) = 1 ∀ s,a

Learning rate = 𝞪

Discounting factor = 𝞬

s1, a1 -> s2

R(s1, a1) = 100

From bellman’s equation,

q’(s1, a1) = 𝞪[R + 𝞬.max(q(s’, a’))] + (1 - 𝞪).q(s1,a1)

q’(s1, a1) = 𝞪[100 + 𝞬.max(q(s2, a’))] + (1 - 𝞪).1

now, q(s2, a’) = 1 ∀ a

=> max(q(s2, a’)) = 1

=> q’(s1, a1) = 𝞪[100 + 𝞬.1] + (1 - 𝞪).1

q’(s1, a1) = 𝞪[100 + 𝞬] + (1 - 𝞪)

q’(s1, a1) = 𝞪[99 + 𝞬] + 1

Rest of the Q-table remains intact.

Thus, the updated values will be

**q(s, a) = 1 ∀ s,a - {s1, A1}**

**q(s1, a1) = 𝞪[99 + 𝞬] + 1**