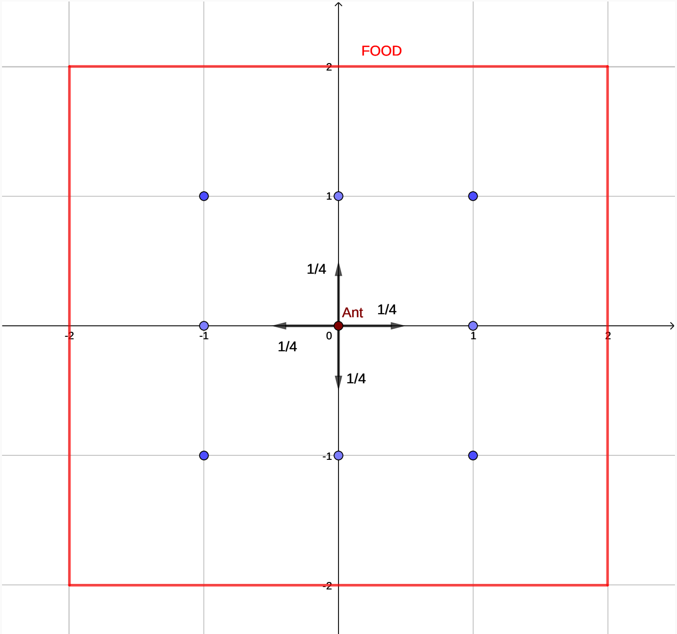
The anthill problem

A gentle introduction to Markov chains and how to compute the average time to reach an absorbant state

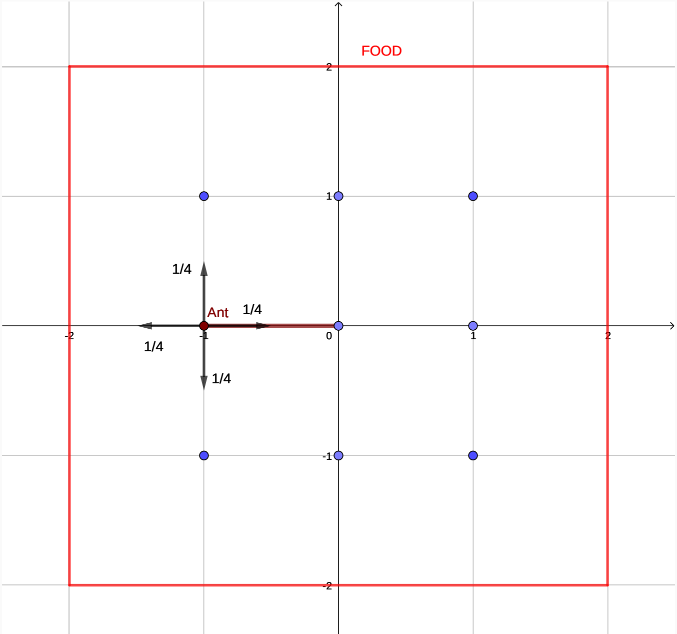
Duy Nghi Benoît Tran

August 04th 2022

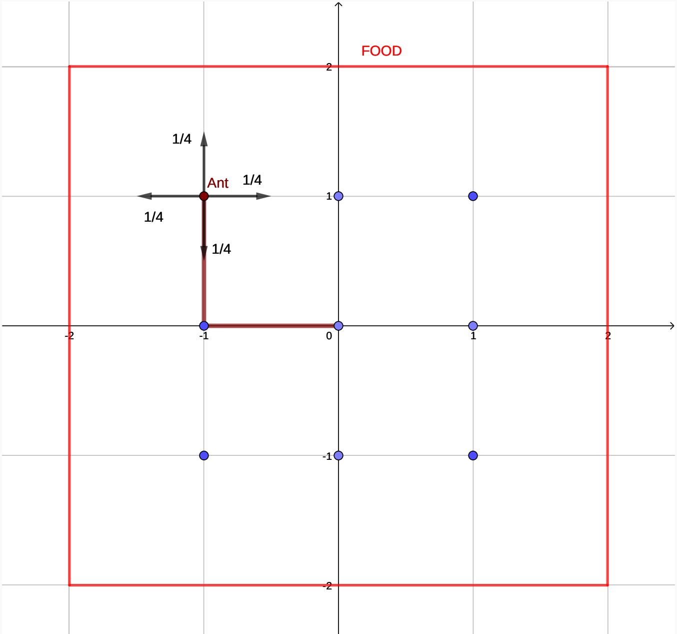
The anthill problem (1/2)



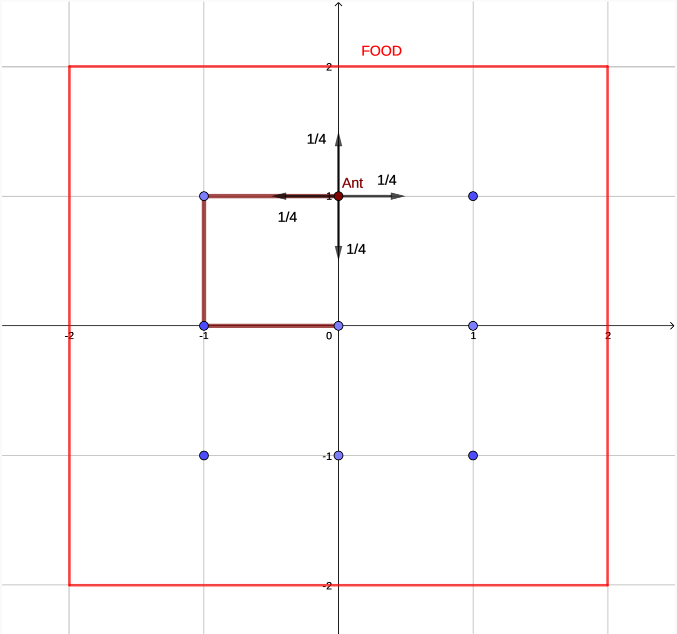
The anthill problem (1/2)



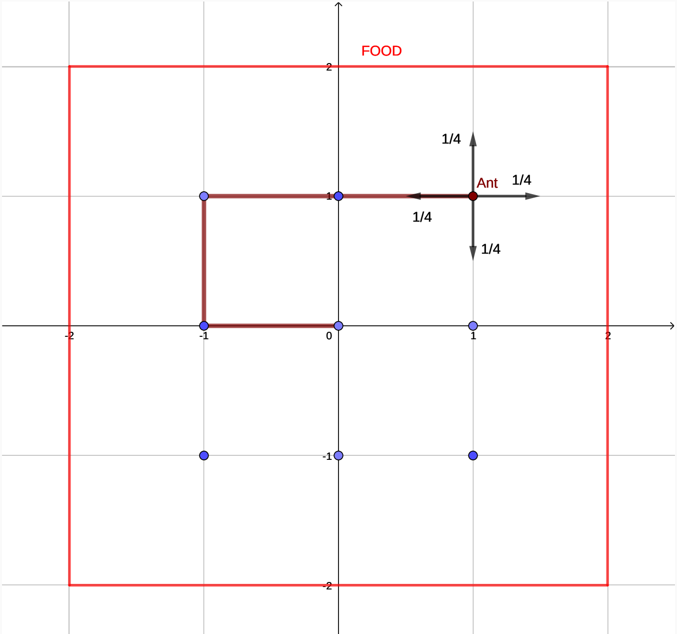
The anthill problem (1/2)



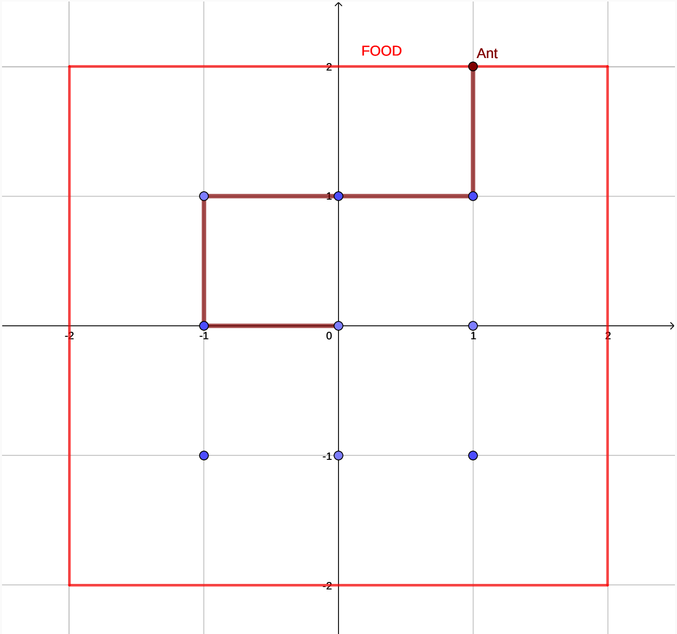
The anthill problem (1/2)



The anthill problem (1/2)



The anthill problem (1/2)



The anthill problem (2/2)

The anthill problem

An ant leaves its anthill in order to forage for food. It moves with the speed of 1 cm per second, but it doesn't know where to go, therefore every second it moves randomly 1 cm directly north, south, east or west with equal probability.

If the food is located on east-west lines 2 cm to the north and 2 cm to the south, as well as on north-south lines 2 cm to the east and 2 cm to the west from the anthill, how long will it take the ant to reach the food on average?

Question

Can you write a simple function (in Julia, Python, R...) which given a number of iterations (e.g. 1000), returns the average number of seconds the ant needed to reach the food?

A straightforward simulation in Python

```
import random

def simulation(iter):
    results = [0]*iter
    for i in range(iter):
        count = 0
        x,y = [0,0]
        while abs(x) < 2 and abs(y) < 2:
            r = random.random()
            count += 1
            if r < 0.25:
                x += 1
            elif r < 0.5:
                x -= 1
            elif r < 0.75:
                y += 1
            else:
                y -= 1
        results[i] = count
    return(sum(results)/iter)

simulation(1000)
```

Guessing the result by simulation (3/3)

Running the previous simulation several times we get outputs like this

$$4.553,\ 4.446,\ 4.507,\ 4.490\ldots$$

Guessing the result by simulation (3/3)

Running the previous simulation several times we get outputs like this

$$4.553,\ 4.446,\ 4.507,\ 4.490\ldots$$

Conjecture

The ant needs 4.5 seconds to reach the food on average.

Guessing the result by simulation (3/3)

Running the previous simulation several times we get outputs like this

$$4.553,\ 4.446,\ 4.507,\ 4.490\ldots$$

Conjecture

The ant needs 4.5 seconds to reach the food on average.

Goal

We will prove our conjecture using the formalism of Markov chains, which gives an intuitive and elegant way of proving the result.

Markov chain (1/6): Matrix and graph representation

Matrix representation of a Markov chain

A Markov chain is characterized by a couple $(\mathcal{X}, P)$ where

- $\mathcal{X} = \{x_1, x_2, \dots, x_N\}$ is a finite set of states,
- $P = (p_{ij})_{1 \leq i,j \leq n}$ is the transition matrix defined by p_{ij} the probability of moving from the state x_i to the state x_j in one step.

Markov chain (1/6): Matrix and graph representation

Matrix representation of a Markov chain

A Markov chain is characterized by a couple $(\mathcal{X}, P)$ where

- $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$ is a finite set of states,
- $P = (p_{ij})_{1 \leq i,j \leq n}$ is the transition matrix defined by p_{ij} the probability of moving from the state x_i to the state x_j in one step.

Graph representation of a Markov chain

A visual representation of a Markov chain is given by a weighted directed graph where

- the set of nodes is the finite set $\mathcal{X} = \{x_1, \ldots, x_N\}$,
- there is an edge with value p_{ij} from x_i to x_j if $p_{ij} > 0$.

Markov chain (2/6): Definition as a stochastic process

Markov chain as a stochastic process

A Markov chain (MC) is a sequence of random variables $X_1, X_2, \ldots, X_n, \ldots$ with

- values in $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, a finite set of states,
- and the Markov property.

Markov chain (2/6): Definition as a stochastic process

Markov chain as a stochastic process

A Markov chain (MC) is a sequence of random variables $X_1, X_2, \ldots, X_n, \ldots$ with

- values in $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, a finite set of states,
- and the Markov property.

Informally, Markov property: future events only depend on the present state of the system. That is, they are independent of past states.

Markov chain (2/6): Definition as a stochastic process

Markov chain as a stochastic process

A Markov chain (MC) is a sequence of random variables $X_1, X_2, \ldots, X_n, \ldots$ with

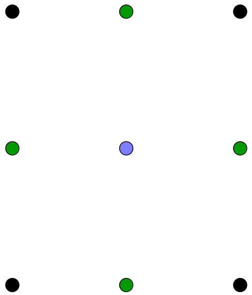
- values in $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, a finite set of states,
- and the Markov property.

Informally, Markov property: future events only depend on the present state of the system. That is, they are independent of past states.

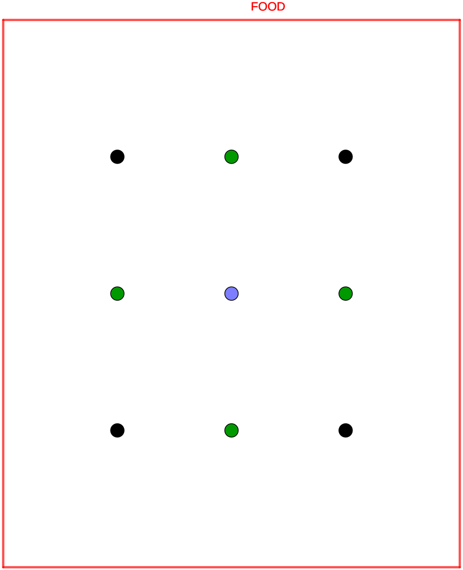
Formally, $(X_i)_{i \geq 1}$ has the Markov property if for every $i \geq 1$ and possible states $x_1, \ldots x_i$ we have

$$\mathbb{P}\left(X_{i+1} = x_{i+1} \mid X_i = x_i, \ldots X_1 = x_1\right) = \mathbb{P}\left(X_{i+1} = x_{i+1} \mid X_i = x_i\right).$$

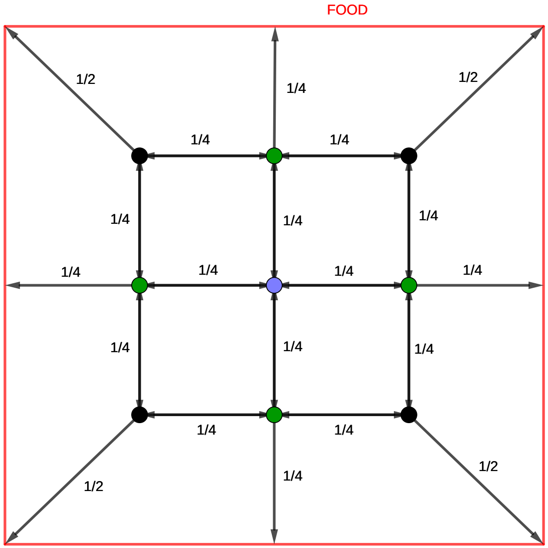
Markov chain (3/6): graph of the anthill problem



Markov chain (3/6): graph of the anthill problem



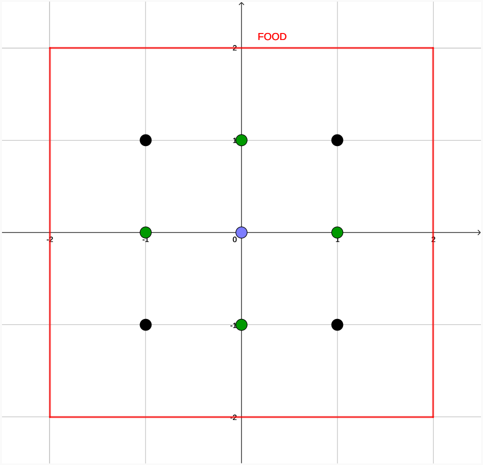
Markov chain (3/6): graph of the anthill problem



Markov chain (4/6): the anthill problem

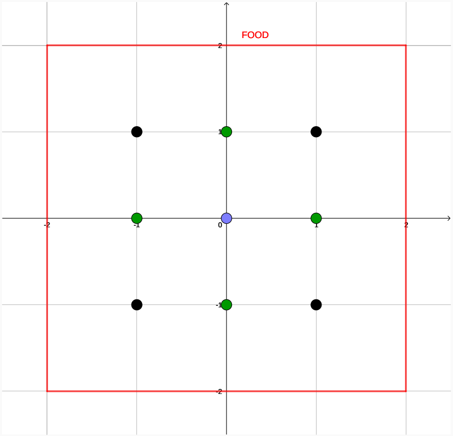
For the anthill problem, one can observe that the problem has many symmetries.

What matters is the distance to the anthill, located at (0,0).



Question

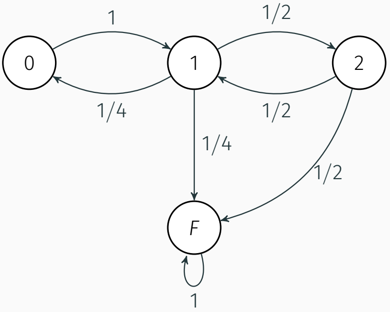
Can you model the anthill problem as a Markov chain where the set of states $\mathcal{X}$ has at most 4 elements? Also give the associated transition matrix P and its associated graph.



Set of states: $\mathcal{X} = \{0, 1, 2, F\}$ distances to the anthill and "food state" F.

Transition matrix $P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1/4 & 0 & 1/2 & 1/4 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$

Graph of the anthill problem



Absorbing Markov chain (1/3)

Absorbing Markov chain

- A state x_i is absorbing if $p_{ii} = 1$.
- A Markov chain is absorbing if from every state, an absorbing state can be reached after a finite number of steps.

Absorbing Markov chain (1/3)

Absorbing Markov chain

- A state x_i is absorbing if $p_{ii} = 1$.
- A Markov chain is absorbing if from every state, an absorbing state can be reached after a finite number of steps.

For an absorbing Markov chain, we first write the non-absorbing states then the absorbing states.

Absorbing Markov chain (1/3)

Absorbing Markov chain

- A state x_i is absorbing if $p_{ii} = 1$.
- A Markov chain is absorbing if from every state, an absorbing state can be reached after a finite number of steps.

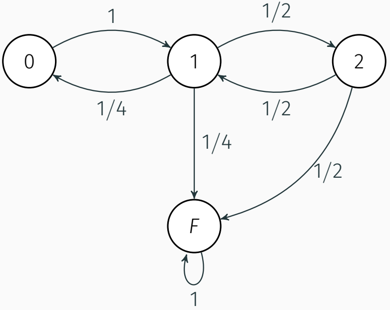
For an absorbing Markov chain, we first write the non-absorbing states then the absorbing states.

We have the following decomposition of the transition matrix

$$P = \begin{pmatrix} Q & R \\ 0 & I \end{pmatrix},$$

where Q square matrix involving the non-absorbing states, R matrix involving the states that can reach the absorbing states.

Absorbing Markov chain (2/3): example of the anthill problem



Transition matrix $P = \begin{pmatrix} Q & R \\ 0 & I \end{pmatrix}$,

where $Q = \begin{pmatrix} 0 & 1 & 0 \\ 1/4 & 0 & 1/2 \\ 0 & 1/2 & 0 \end{pmatrix}$ and $R = \begin{pmatrix} 0 \\ 1/4 \\ 1/2 \end{pmatrix}$.

Absorbing Markov chain (3/3): some properties

Let x_i and x_j be two non-absorbing states.

- The probability of reaching x_j starting from x_i after n steps is equal to the coefficient (i, j) of Q^n.

Absorbing Markov chain (3/3): some properties

Let x_i and x_j be two non-absorbing states.

- The probability of reaching x_j starting from x_i after n steps is equal to the coefficient (i, j) of Q^n.
- The matrix $I - Q$ is invertible and we have

$$I + Q + Q^2 + \ldots + Q^n + \ldots = (I - Q)^{-1} =: N.$$

Absorbing Markov chain (3/3): some properties

Let x_i and x_j be two non-absorbing states.

- The probability of reaching x_j starting from x_i after n steps is equal to the coefficient (i, j) of Q^n.
- The matrix $I - Q$ is invertible and we have

$$I + Q + Q^2 + \ldots + Q^n + \ldots = (I - Q)^{-1} =: N.$$

- N is called the fundamental matrix of the absorbing Markov chain.
  The coefficient (i, j) of N is equal to the average number of times the state x_j is attained starting from x_i.

Solution to the anthill problem

Writing T the average waiting time until the ant reaches food,

$$T = \sum_{j=1}^{3} N_{1j}.$$

Solution to the anthill problem

Writing T the average waiting time until the ant reaches food,

$$T = \sum_{j=1}^{3} N_{1j}.$$

Proof.

Recall: coefficient (i, j) of N is equal to the average number of times the non-absorbing state x_j is attained starting from x_i.

$\square$

Solution to the anthill problem

Writing T the average waiting time until the ant reaches food,

$$T = \sum_{j=1}^{3} N_{1j}.$$

Proof.

Recall: coefficient (i, j) of N is equal to the average number of times the non-absorbing state x_j is attained starting from x_i.

Thus, the sum of the first line of N is equal to the average number of time the ant remains in a non-absorbing state starting from the first state (the center),

$\square$

Solution to the anthill problem

Writing T the average waiting time until the ant reaches food,

$$T = \sum_{j=1}^{3} N_{1j}.$$

Proof.

Recall: coefficient (i, j) of N is equal to the average number of times the non-absorbing state x_j is attained starting from x_i.

Thus, the sum of the first line of N is equal to the average number of time the ant remains in a non-absorbing state starting from the first state (the center),

which is equal to the average time until the absorbing state is reached. $\square$

Solution to the anthill problem

Computing the solution

Write a short program which computes the sum $\sum_{j=1}^{3} N_{1j}$.

Solution to the anthill problem

Computing the solution

Write a short program which computes the sum $\sum_{j=1}^{3} N_{1j}$.

A code in Python

```
import numpy as np
Q = np.array([[0,    1,   0   ],
              [0.25, 0,   0.5 ],
              [0,    0.5, 0   ]])
N = np.linalg.inv(np.identity(3) - Q)
T = sum(N[0])
print(T)
```

Solution to the anthill problem

Computing the solution

Write a short program which computes the sum $\sum_{j=1}^{3} N_{1j}$.

A code in Python

```
import numpy as np
Q = np.array([[0,    1,   0   ],
              [0.25, 0,   0.5 ],
              [0,    0.5, 0   ]])
N = np.linalg.inv(np.identity(3) - Q)
T = sum(N[0])
print(T)
```

The ant reaches the food after 4.5 seconds on average.

Have you understood? Example of the Drunkard walk

The Drunkard walk

A drunkard walks along a road. At each intersection, noted 1, 2 and 3, he changes direction with probability 1/2. His house is at the beginning of the road and a bar is at the end of the road.

If he reaches either the bar or his home, he stays there the whole night. He starts at intersection 2.

Have you understood? Example of the Drunkard walk

The Drunkard walk

A drunkard walks along a road. At each intersection, noted 1, 2 and 3, he changes direction with probability 1/2. His house is at the beginning of the road and a bar is at the end of the road.

If he reaches either the bar or his home, he stays there the whole night. He starts at intersection 2.

Question 1

What are the set of states, transition matrix and graph of the Drunkard walk?

Have you understood? Example of the Drunkard walk

The Drunkard walk

A drunkard walks along a road. At each intersection, noted 1, 2 and 3, he changes direction with probability 1/2. His house is at the beginning of the road and a bar is at the end of the road.

If he reaches either the bar or his home, he stays there the whole night. He starts at intersection 2.

Question 1

What are the set of states, transition matrix and graph of the Drunkard walk?

Question 2

What is the average number of steps until the drunkard reaches either his home or the bar?

The Drunkard walk - solution of Question 1

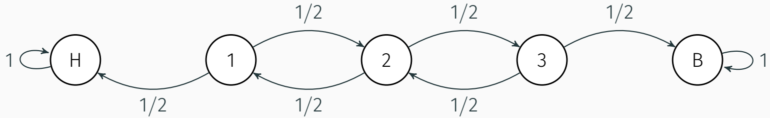
Set of states

$\mathcal{X} = \{1, 2, 3, H, B\}$, H for Home and B for Bar.

Transition matrix

$$P = \left(\begin{array}{ccc|cc} 0 & 1/2 & 0 & 1/2 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 1/2 \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{c|c} Q & R \\ \hline 0 & I \end{array}\right).$$

Graph of the Drunkard walk



The Drunkard walk - solution of Question 2

1) Compute the fundamental matrix

2) Sum the right coefficients

```
import numpy as np
Q = np.array([[0,   0.5, 0  ],
              [0.5, 0,   0.5],
              [0,   0.5, 0  ]])
N = np.linalg.inv(np.identity(3) - Q)
T = sum(N[1])
print(T)
```

The output is 4.0.

3) Check by simulation

```
import random

def simulation_drunkard(iter):
    results = [0]*iter
    for i in range(iter):
        count = 0
        x = 2
        while x < 4 and x > 0:
            r = random.random()
            count += 1
            if r < 0.5:
                x += 1
            else:
                x -= 1
        results[i] = count
    return(sum(results)/iter)

simulation_drunkard(1000)
```

The drunkard reaches the bar or his house after 4 steps on average.