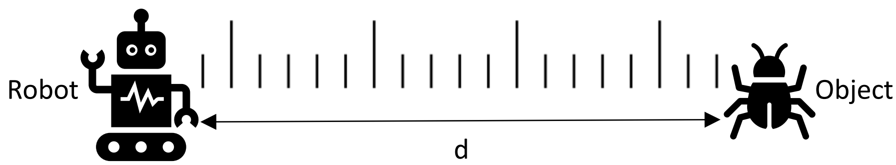
EAS-502 Project 1

Team Members:

1. Nabeel Khan 56
2. Vijaya Rana 106



The **experiment starts at t = 0 (d>0)**. Let ***T*** be the time at which the distance between Robot and object is zero (d=0). Experiment ends when the **robot and the object land on the same spot, or when they cross each other**.

We define:
A_d: *The event that* ***initially*** *the robot and the object are* ***d*** *units apart*
B_d: *The event that* ***after one second*** *the robot and object are* ***d*** *units apart*

Strategy 1: At each unit of time, the robot will move 1 unit toward the object regardless of the object movement.

Analytical Solution

For the initial distance d there are 3 cases which are mentioned below considering robot moves 1 unit towards object:

Case	Object movement	Robot movement	Distance	Probability
1	Left	Right	d-2	p
2	Right	Right	d	p
3	Stationary	Right	d-1	1-2p

Where p = Probability of movement of Object

Breaking $E[T|A_d]$ by using Total expectation theorem for above 3 cases:

$E[T|A_d] = P(B_d|A_d)E[T|A_d \cap B_d] + P(B_{d-1}|A_d)E[T|A_d \cap B_{d-1}] + P(B_{d-2}|A_d)E[T|A_d \cap B_{d-2}]$ → Equation 1

For simplification considering: $E[T|A_d] = E_d$ & P_{OM}=p

Simplifying further Equation 1

$E_d = p(1+E_d) + (1-2p)(1+E_{d-1}) + p(1+E_{d-2})$

Rearranging for E_d

$\boxed{E_d = (p + (1-2p)(1+E_{d-1}) + p(1+E_{d-2}))/(1-p)}$ → This is *General term for the Strategy 1*

We calculate E[T|A_1] using the ***General term for the Strategy 1*** and then we used this result to compute E[T|A_2]. We'll use MATLAB to calculate E[T|A_d] for the relevant values of d.

E_1 = 1-p+p (Ed) + p

E_2 =(2-3p)/((1-p)^2)

Now using this we can calculate E_3 to E_{19}

MATLAB Code for Strategy 1:

```
function [final_expected_vals] = strategy_1()
% Function to calculate expected values of time for P_om from .1 to .5

    final_expected_vals = [];

    % Using P_om from .1 to .5
    for p_om = (1:5)/10

        % Starting with the E1 and E2 calculated manually using general term
        exp_vals = [1/(1-p_om), (2-3*p_om)/((1-p_om)*(1-p_om))];

        % Calculating E3 to E19 using general equation
        for d = 3:19
            E_d = (p_om + (1-2*p_om)*(1+ exp_vals(d-1)) + p_om*(1+ exp_vals(d-2)))/(1-p_om);
            exp_vals(d) = E_d;
        end

        % Now calculating the expected value using conditional expectation
        % Using pmf of d (uniform distribution with Pd = 0.1)
        expected_total = 0;
        for d = 10:19
            expected_total = expected_total + 0.1*exp_vals(d);
        end

        disp(expected_total);
        final_expected_vals(end + 1) = expected_total;
    end
end
```

Analytical Result for Strategy 1:

Probability of Object Movement (P_{OM} or p)	Expected Value of Time (E_T)
0.1	14.60
0.2	14.70
0.3	14.80
0.4	14.90
0.5	15.00

Strategy 3: At each unit of time, the robot will move 1 unit toward the object if the object moves to either left or right, and the robot will stop if the object stops.

Analytical Solution:

For the initial distance d there are 3 cases which are mentioned below considering robot moves 1 unit towards object:

Case	Object movement	Robot movement	Distance	Probability
1	Left	Right	d-2	p
2	Right	Right	d	p
3	Stationary	Stationary	d	1-2p

Where p = Probability of movement of Object

Breaking $E[T|A_d]$ by using Total expectation theorem for above 3 cases:

$E[T|A_d] = P(Object\ moves\ right\ )E[T|A_d \cap B_d] + P(Object\ stationary)E[T|A_d \cap B_d] + P(Object\ moves\ left)E[T|A_d \cap B_{d-2}]$

For simplification considering $E[T|A_d]$ = E_d & P_{OM}=p

Simplifying further:

$E_d = p\ (1+ E_{d-2}) + p(1+E_d) + (1-2p)\ (1+E_d)$

Now after solving for E_d, we get

$\boxed{E_d = (1+pE_{d-2})/p}$ ➔ This is *General term for the Strategy 3*

We calculate E[T|A_1] using the *General term for the Strategy 3* and then we used this result to compute E[T|A_2].
We'll use MATLAB to calculate E[T|A_d] for the relevant values of d.

$E_1 = 1/p$

$E_2 = 1/p$

Now using this we can calculate E_3 to E_{19}

MATLAB Code for Strategy 3:

```
function [final_expected_vals] = strategy_3()
% Function to calculate expected values of time for P_om from .1 to .5

    final_expected_vals = [];

    % Using P_om from .1 to .5
    for p_om = (1:5)/10

        % Starting with the E1 and E2 calculated manually
        exp_vals = [1/(p_om), 1/(p_om)];

        % Calculating E3 to E19 using general term
        for d = 3:19
            E_d = (1 + (p_om)*(exp_vals(d-2)))/(p_om);
            exp_vals(d) = E_d;
        end

        % Now calculating the expected value using conditional expectation
        % Using pmf of d (uniform distribution with Pd = 0.1)
        expected_total = 0;
        for d = 10:19
            expected_total = expected_total + 0.1*exp_vals(d);
        end

        disp(expected_total);
        final_expected_vals(end + 1) = expected_total;
    end
end
```

Analytical Result for Strategy 3:

Probability of Object Movement (P_{OM} or p)	**Expected Value of Time (E_T)**
0.10	75.00
0.20	37.50
0.30	25.00
0.40	18.75
0.50	15.00

Strategy 2*: At each unit of time, the robot will move 1 unit toward the object (with the probability of 1.5 P_{OM} or will stop (with the probability of 1-1.5P_{OM}) regardless of the object movement.

For the initial distance d there are 6 cases which are mentioned below considering robot moves 1 unit towards object:

Case	Robot movement	Object movement	Distance	Probability
1	Still	Left	d-1	(1-q)p
2	Still	Right	d+1	(1-q)p
3	Still	Still	d	(1-q)(1-2p)
4	Right	Left	d-2	qp
5	Right	Right	d	qp
6	Right	Still	d-1	q(1-2p)

Where p = Probability of movement of Object, and q = Probability of movement of robot

$E[T|A_d] = P(Robot\ still, Object\ moves\ Left)E[T|A_d \cap B_{d-1}] + P(Robot\ still, Object\ moves\ Right)E[T|A_d \cap B_{d+1}] + P(Robot\ still, Object\ Still)E[T|A_d \cap B_d] + P(Robot\ moves\ right, Object\ moves\ Left)E[T|A_d \cap B_{d-2}] + P(Robot\ moves\ right, Object\ moves\ Right)E[T|A_d \cap B_d] + P(Robot\ moves\ right, Object\ Still)E[T|A_d \cap B_{d-1}]$

Simplifying above equation considering $E[T|A_d]$ = E_d

$E_{d=}$(1+(p+q-3pq) E_{d-2}+(3 pq-2p-q) E_{d-1}+(pq)E_{d-3})/p(q-1)

Where q=1.5p, solving we get

E_d = (1+(2.5p-4.5p^2)E_{d-2} + ((4.5*p^2) - 3.5*p)E_{d-1}+(1.5 p^2) E_{d-3}))/p(1.5p- 1) ➔ This is *General term for the Strategy 2*

Using *General term for the Strategy 2* we calculate E_1, E_2, E_3 for respective values of p shown in table

	E_1	E_2	E_3
p=0.1	7.109830201	13.74703778	20.41566273
p=0.2	3.832148359	7.090836764	10.4353403
p=0.3	2.796251249	4.870194275	7.130718951
p=0.4	2.349754291	3.736455736	5.51786687
p=0.5	2.194335081	2.971675407	4.664041954

MATLAB Code for Strategy 2:

```
function [final_expected_vals] = strategy_2()
% Function to calculate expected values of time for P_om from .1 to .5

    initial_E = [7.109830200957157  13.747037779905098 20.415662725586326
                 3.8321483594592816 7.090836763705904  10.435340300715005
                 2.7962512487710653 4.870194275499013  7.130718950732399
                 2.3497542908541056 3.736455736129949  5.5178668695576
                 2.1943350814194544 2.9716754070972726 4.664041954066908];

    final_expected_vals = [];

    % Using P_om from .1 to .5
    for iter_count = (1:5)

        % Using P_om from .1 to .5
        p_om = iter_count/10;

        % Starting with the E1, E2, and E3 calculated manually
        E = initial_E(iter_count,:);

        % Calculating E4 to E19 using general equation
        for d = 4:19
            E(d)=(1+(2.5*p_om-4.5*p_om^2)*E(d-2)+((4.5*p_om^2)-3.5*p_om)*E(d-1)
            +(1.5*(p_om*^2)*E(d-3)))/(p_om*(1.5*p_om - 1));
        end

        % Now calculating the expected value using conditional expectation
        % Using pmf of d (uniform distribution with Pd = 0.1)
        expected_total = 0;
        for d = 10:19
            expected_total = expected_total + 0.1*E(d);
        end

        disp(expected_total);
        final_expected_vals(end + 1) = expected_total;
    end
end
```

Analytical Result for Strategy 2:

Probability of Object Movement (P_{OM} or p)	**Expected Value of Time (E_T)**
0.10	97.082
0.20	48.767
0.30	32.678
0.40	24.651
0.50	19.856

Conclusion

We used simulation for 10000 iterations to **arrive at the same results** as concluded in the table below

Strategy	P_om	Analytical Expected Value	Simulation		
			Iterations	Total Time	Simulated Expected Value
Stg_1	0.10	14.600	10000	146042	14.6042
Stg_1	0.20	14.700	10000	146760	14.676
Stg_1	0.30	14.800	10000	147276	14.7276
Stg_1	0.40	14.900	10000	149199	14.9199
Stg_1	0.50	15.000	10000	149503	14.9503
Stg_2	0.10	97.082	10000	967419	96.7419
Stg_2	0.20	48.767	10000	489432	48.9432
Stg_2	0.30	32.678	10000	327241	32.7241
Stg_2	0.40	24.651	10000	245996	24.5996
Stg_2	0.50	19.856	10000	198819	19.8819
Stg_3	0.10	75.000	10000	749603	74.9603
Stg_3	0.20	37.500	10000	375690	37.569
Stg_3	0.30	25.000	10000	251017	25.1017
Stg_3	0.40	18.750	10000	188280	18.828
Stg_3	0.50	15.000	10000	150856	15.0856

Python Code for simulation

```
import numpy as np

import random

import pandas as pd


table_results = []


for p_om in [.1, .2, .3, .4, .5]:

    robot_position = 0

    obj_position = np.random.choice(range(10,20))

    robot_mvmts = np.array([0,1])

    obj_mvmts = np.array([-1,0,1])

    obj_mvmt_probs = np.array([p_om, 1-2*p_om, p_om])

    robot_mvmt_probs = np.array([1-1.5*p_om, 1.5*p_om])


    # strategy 1

    stg_1_times = []
```

```
    for sim_count in range(10000):
        time_count = 0
        while robot_position < obj_position:
            robot_position += 1
            obj_position +=  np.random.choice(obj_mvmts, 1,p=obj_mvmt_probs)[0]
            time_count += 1
        robot_position = 0
        obj_position = np.random.choice(range(10,20))
        stg_1_times.append(time_count)
    print(sum(stg_1_times)/len(stg_1_times))
    table_results.append(['Stg_1', p_om,sum(stg_1_times) ])

    # strategy 2
    stg_2_times = []
    for sim_count in range(10000):
        time_count = 0
        while robot_position < obj_position:
            robot_position += np.random.choice(robot_mvmts,
1,p=robot_mvmt_probs)[0]
            obj_position +=  np.random.choice(obj_mvmts, 1,p=[p_om, 1-2*p_om,
p_om])[0]
            time_count += 1
        robot_position = 0
        obj_position = np.random.choice(range(10,20))
        stg_2_times.append(time_count)
    print(sum(stg_2_times)/len(stg_2_times))
    table_results.append(['Stg_2', p_om,sum(stg_2_times) ])

    # strategy 3
    stg_3_times = []
    for sim_count in range(10000):
        time_count = 0
        while robot_position < obj_position:
            obj_movement = np.random.choice([-1,0,1], 1,p=[p_om, 1-2*p_om,
p_om])[0]
```

```
        obj_position += obj_movement
        robot_position += abs(obj_movement)
        time_count += 1
    robot_position = 0
    obj_position = np.random.choice(range(10,20))
    stg_3_times.append(time_count)
print(sum(stg_3_times)/len(stg_3_times))
table_results.append(['Stg_3', p_om,sum(stg_3_times) ])
```