# AI ASSIGNMENT 3 — Uncertainty, Bayesian Nets, HMM and Kalman Filtering

Ritika Thakur | 2022408

## Theory

# AI ASSIGNMENT-3

RITIKA THAKUR (2022408)

Q1. (a) 1. Direct Sampling :-

Strengths = effective for straightforward marginal probabilities like "About 70% of people travel for either leisure or business.", "About 80% of people prefer to travel by air or train."
= Simple to implement (when dependencies aren't complex)

Weaknesses = difficult to handle conditional probabilities
= may waste computation on rare events.

2. Rejection Sampling :-

Strengths = good for conditional probabilities in the dataset
= could effectively sample rare events like air travel with low stress.

Weaknesses = might have high rejection rates

3. Gibbs Sampling :-

Strengths = good for handling multiple interrelated variables
= can mantain conditional relationships in data.

Weaknesses = requires complex implementation
= could get stuck in local nodes of the distribution.

⇒ Gibbs is the most appropriate method for our dataset.

(b) P(leisure | train) = 0.4, sample size = 100
# people preferring train = 30

E[no. of leisure travellers amongst train travellers]
= 30 × 0.4 = 12

(c) $P(air) = 0.8$

$P(business/air) = 0.2$

$\rightarrow P(air \wedge business) = P(air) \cdot P(business/air)$

$= 0.8 \times 0.2$

$= 0.16$

(d) (1) Effects on accuracy of an increased sample size :-

→ reducing sampling error

→ sample mean converges closer to true mean.

(2) Effects on precision of an increased sample size :-

→ smaller standard error

→ narrower confidence intervals.

(3) Tradeoffs :-

→ more data requires more time to collect

→ higher cost of data computation

→ critical probabilities might need larger samples

→ less critical measures might be adequate with smaller samples.

→ Implications for the given dataset.

Q2.

(a) Let B : person reads books

A : person accesses academic journals

C : person participates in book clubs

Statement 1: $P(B \vee A) = 0.91$

Statement 2: $P(A/B) = 0.40$

$P(B \wedge A') = 0.60$

Statement 3: $P(C|B) = 0.32$

Statement 4: $P(A \wedge B') = 0.227$

Statement 5: $P(A' \wedge B') = 0.090$

Statement 6: $P(A|B') = 0.716$

Statement 7: $P(C \wedge A) = 0.088$

Statement 8: $P(C \vee A) = 0.631$

Statement 9: $P(A|C) = 0.40$

Statement 10: $P(A) = 0.50$

Statement 11: $P(C|B') = 0.0044$

(b) 3 main axioms are :-

1. Non-negativity : $P(x) \in [0, 1]$
2. Unity : $\sum_{x \in X} P(x) = 1$
3. Additivity : $P(A \vee B) = P(A) + P(B)$ ; $P(A \wedge B) = \phi$

Verification :-

1. All probabilities are within the range $[0, 1]$.
2. $P(A' \wedge B') + P(A \vee B) = 1$
   i.e. normalisation property is satisfied.
   From the given propositions we can only verify these two probabilities for the given property.
3. Since A, B, C are not mutually disjoint events thus we do not check for axiom 3.

Hence, from the given propositions we can conclude that they form a valid probability distribution.

(c) Given,

$P(A) = 0.50$ $\quad P(A|B) = 0.40$

$P(A \wedge B') = 0.227$ $\quad P(B \wedge A') = 0.60$

$P(A' \wedge B') = 0.090$ $\quad P(A|B') = 0.716$

$P(C \wedge A) = 0.088$ $\quad P(A|C) = 0.40$

$P(B \vee A) = 0.91$ $\quad P(C|B) = 0.32$

$$\Rightarrow P(B) = P(B \vee A) - P(A \wedge B')$$
$$= 0.91 - 0.227$$
$$= 0.683$$

$$P(A \wedge B) = P(A) - P(A \wedge B')$$
$$= 0.5 - 0.227$$
$$= 0.273$$

$$P(C \wedge B) = P(C|B)\,P(B) = 0.32 \times 0.683$$
$$= 0.218$$

$P(C \vee A) = 0.631$

$P(C \wedge B) = 0.218$

$$P(C \wedge B') = P(C \vee A) + P(C \wedge A) - P(C \wedge B) - P(A)$$
$$= 0.631 + 0.088 - 0.218 - 0.5$$
$$= 0.001$$

| Event | $P(C \wedge A)$ | $P(C \wedge A')$ | $P(C)$ | $P(C')$ |
|---|---|---|---|---|
| B | 0.088 | 0.218 | 0.306 | 0.273 |
| B' | 0.227 | 0.001 | 0.228 | 0.090 |
| Total | 0.315 | 0.219 | 0.534 | 0.363 |

(d) So, A, B and C are not conditionally independent.
This can be proved as follows:

B and A are conditionally independent if for given C

$$P(A \wedge B/C) = A(A/C)\, P(B/C)$$

$$P(B \wedge A/C) = \frac{0.088}{0.534} = 0.164$$

$$P(B/C)\, P(A/C) = \frac{0.306}{0.534} \times 0.40 = 0.269$$

$\Rightarrow$ B and A are not cond. independent.

For A and C:

$$P(A \wedge C/B) = \frac{0.088}{0.683} = 0.129$$

$$P(A/B)\, P(C/B) = 0.4 \times 0.32 = 0.128$$

$\Rightarrow$ A and C are not cond. independent.

Q3. A: adversarial perturbations occurred
B: Backdoor attack occurred
M: Misclassification alarm raised.

We want $P(A/M)$

$$P(A/M) = P(M/A) \cdot \frac{P(A)}{P(M)}$$

Assuming initially independent,

$$P(A \cap B) = P(A)\,P(B)$$

$$\Rightarrow P(M) = P(M/A)\,P(A) + P(M/B)\,P(B)$$

$\Rightarrow$ With increased backdoor trigger prevelance, $P(B)$ increases and thus, $P(M)$ and consequently $P(A/M)$ changes (decreases)

$P(A/M)$: prob. of adversarial perturbations given the misclassification alarm

$P(M/A)$: prob. of misclassification alarm given adversarial perturbations.

$P(A)$: prior probab. of adversarial perturbations

$P(M)$: prior probab of misclassification alarm

$P(B)$: prior probab. of backdoor attack

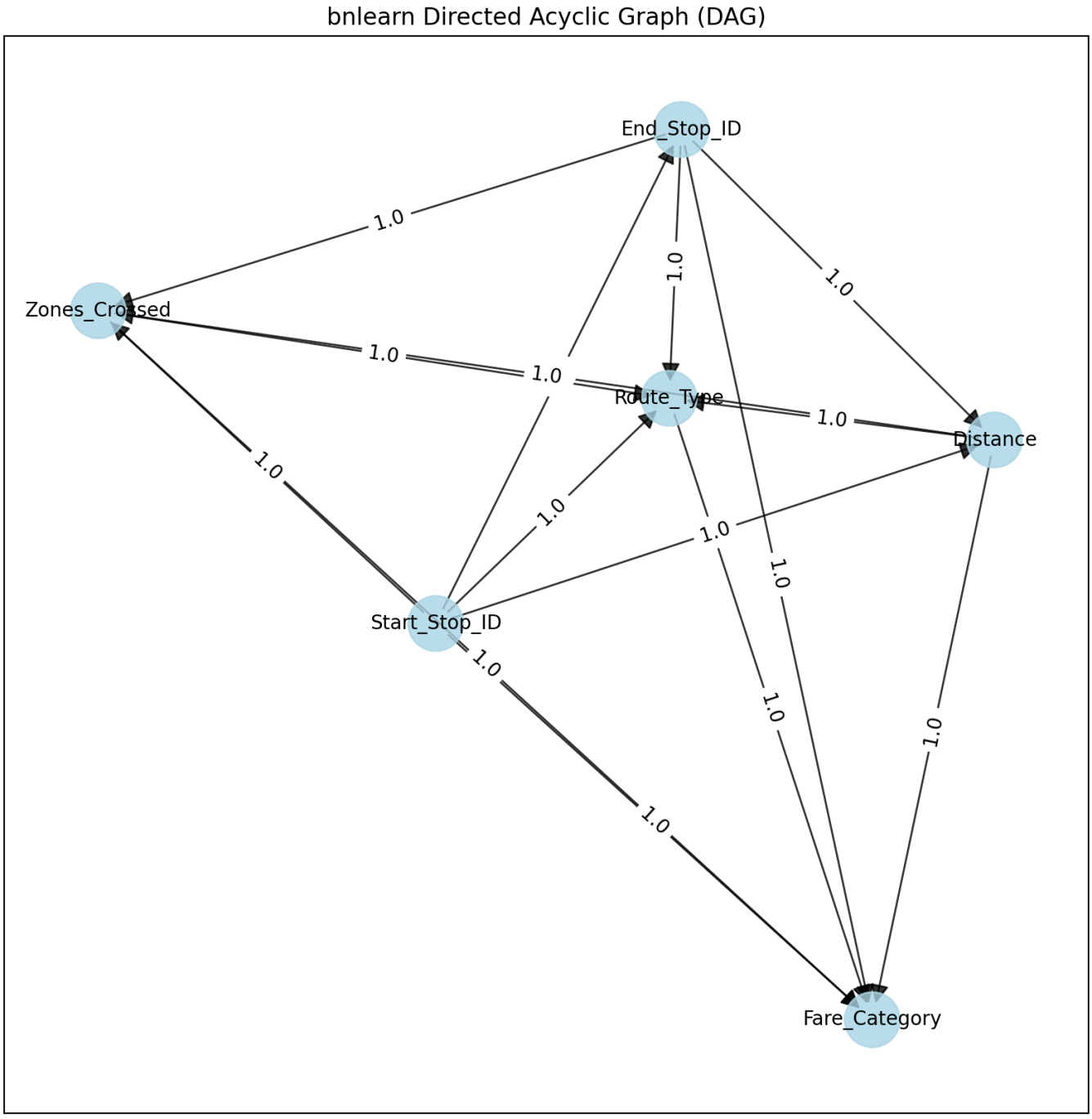
$P(M/B)$: prob. of misclassification alarm given backdoor attacks.

# Coding

## Question 1

**Task 1: Construct the initial Bayesian Network (A) for fare classification.**

Created edges between all possible feature pairs as below:




We see that the degree of each node is 5.

The accuracy obtained was 100%.

```
+----+-----------------+-------------+
Total Test Cases: 350
Total Correct Predictions: 350 out of 350
Model accuracy on filtered test cases: 100.00%
```

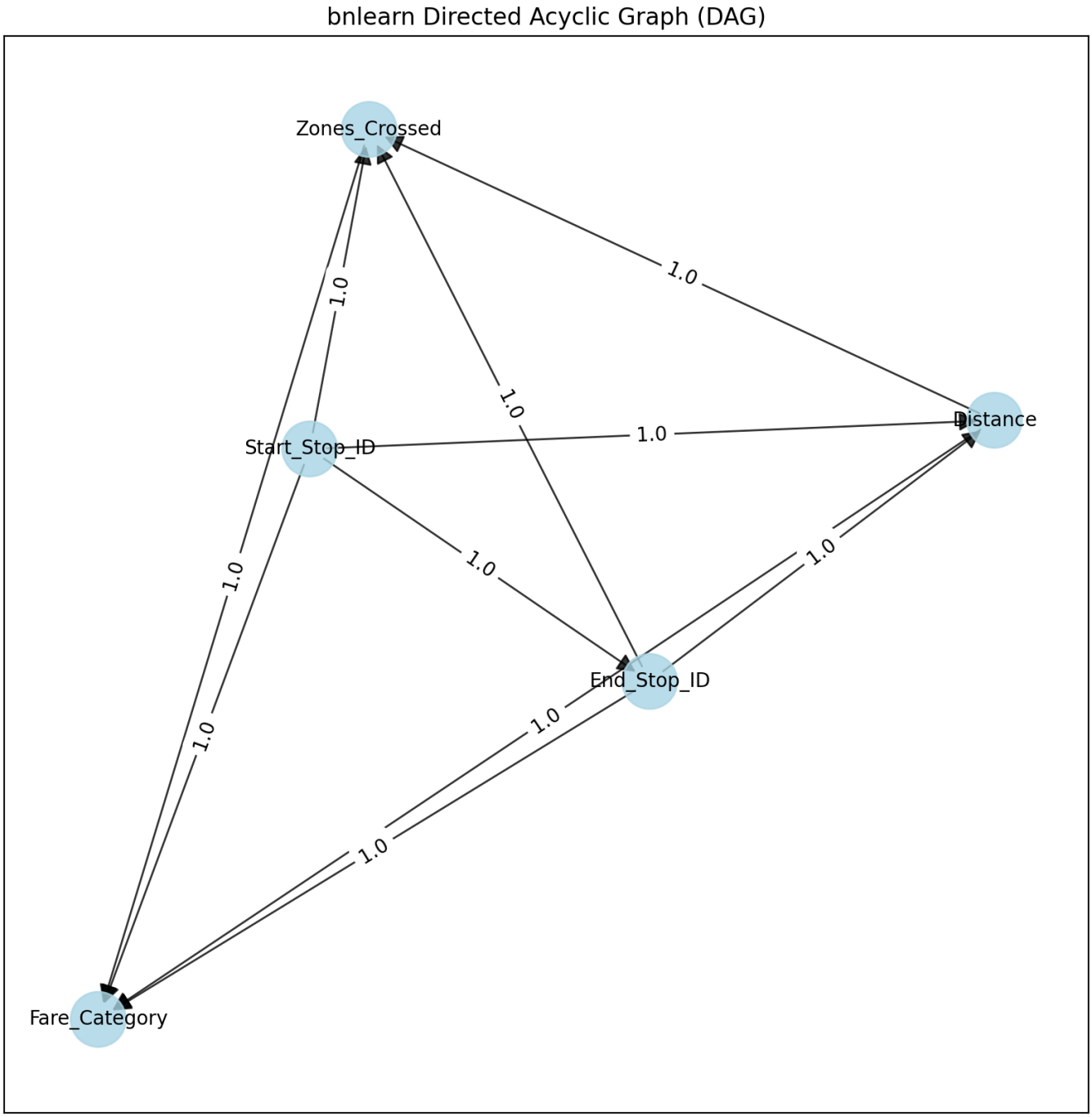
Below is the adjacency matrix of the Bayesian Network depicting the dependencies between the nodes.

| target | Start_Stop_ID | End_Stop_ID | Distance | Zones_Crossed | \ |
|---|---|---|---|---|---|
| source | | | | | |
| Start_Stop_ID | False | False | False | False | |
| End_Stop_ID | False | False | False | False | |
| Distance | True | False | False | False | |
| Zones_Crossed | False | False | False | False | |
| Route_Type | False | False | False | False | |
| Fare_Category | False | True | True | False | |

| target | Route_Type | Fare_Category |
|---|---|---|
| source | | |
| Start_Stop_ID | False | False |
| End_Stop_ID | False | False |
| Distance | False | False |
| Zones_Crossed | False | True |
| Route_Type | False | False |
| Fare_Category | False | False |

**Task 2: Prune the initial Bayesian Network (A) to enhance performance.**

Edge pruning was done by removing edges between nodes using the independence test. The independence test was performed using the chi-square test. The chi-square test was performed between each pair of nodes. If the p-value was greater than the threshold, the edge was removed. The threshold was set to `0.5`.




We see that the maximum degree has reduced to 4.

The accuracy obtained remained 100%.

```
+----+-----------------+-------------+
Total Test Cases: 350
Total Correct Predictions: 350 out of 350
Model accuracy on filtered test cases: 100.00%
```

We can see from the time analysis below that time taken for pruning the network is almost similar to the time taken to create the base network. Also from the snapshot of my jupyter notebook cells we can see that the time taken for evaluating pruned network is lesser than that for base. This is because the number of edges has reduced, and hence the number of computations has reduced.

```
Base Model: Time = 6204.13s, Memory = 13891020.48KB
Pruned Model: Time = 6446.32s, Memory = 13890910.26KB
```

```
train_df, val_df = load_data()

base_model = make_network(train_df.copy())
save_model("base_model.pkl", base_model)
```

✓ 39m 44.2s

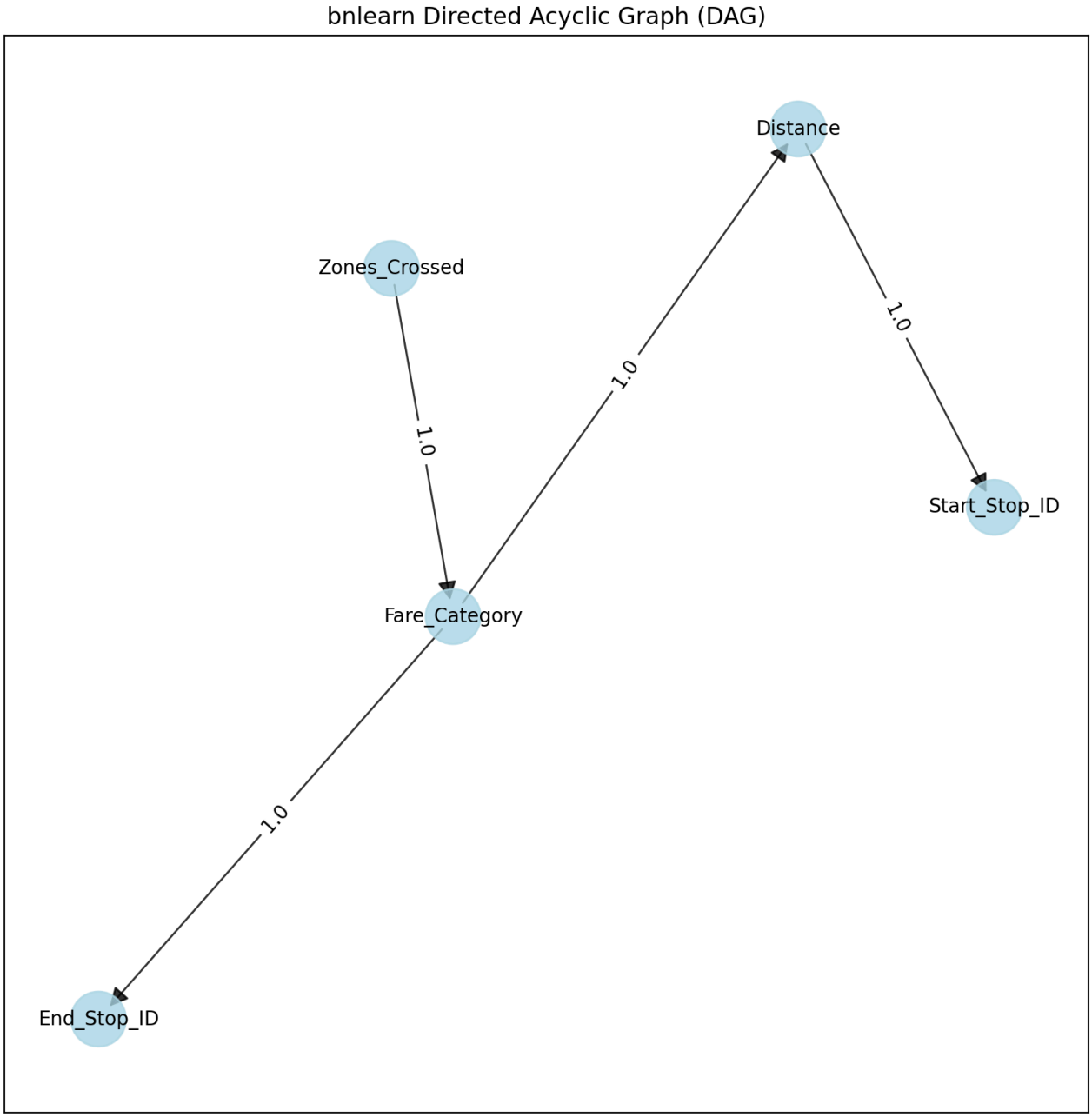
```
pruned_model = make_pruned_network(train_df.copy())
save_model("pruned_model.pkl", pruned_model)
```

✓ 32m 11.3s

**Task 3: Optimize the Bayesian Network (A) by adjusting parameters or using structure refinement methods.**

The following optimization techniques were used:

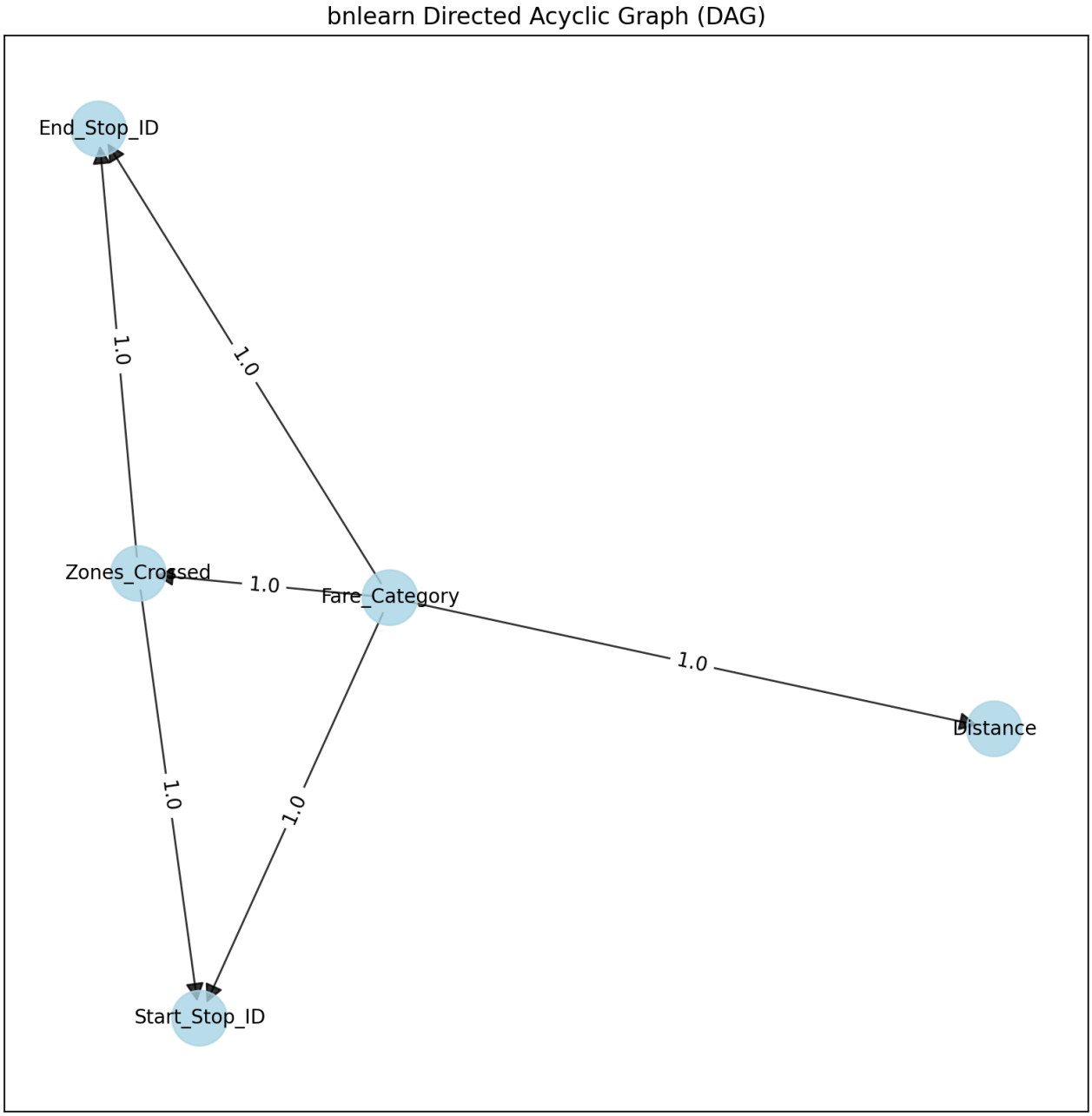
1. **Hill Climb Search**: The hill climb search algorithm was used to optimize the Bayesian network. HillClimbSearch implements a greedy local search that starts from the DAG "start" (default: disconnected DAG) and proceeds by iteratively performing single-edge manipulations that maximally increase the score. The search terminates once a local maximum is found.




Time taken using Hill Climb Search Optimization:

```
Optimized Model (Hill Climbing): Time = 8.11s, Memory = 505.16KB
```

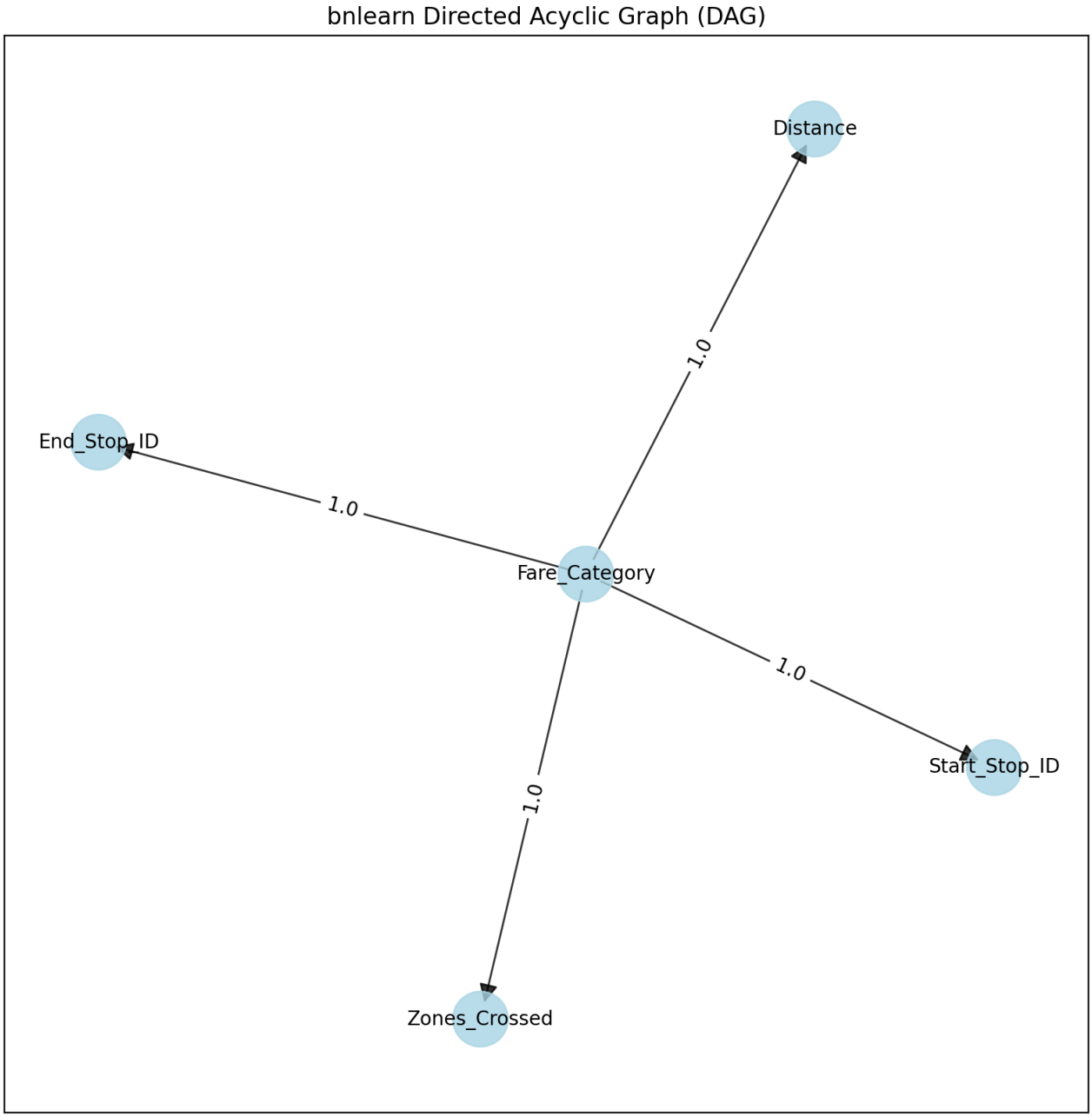
2. **TAN Algorithm**: The Tree-Augmented Naive Bayes (TAN) algorithm was used to optimize the Bayesian network. It relaxes the naive Bayes attribute independence assumption by employing a tree structure, in which each attribute only depends on the class and one other attribute.




Time taken using TAN Algorithm Optimization:

```
Optimized Model (Tree-augmented Naive Bayes): Time = 5.60s, Memory = 16589.59KB
```

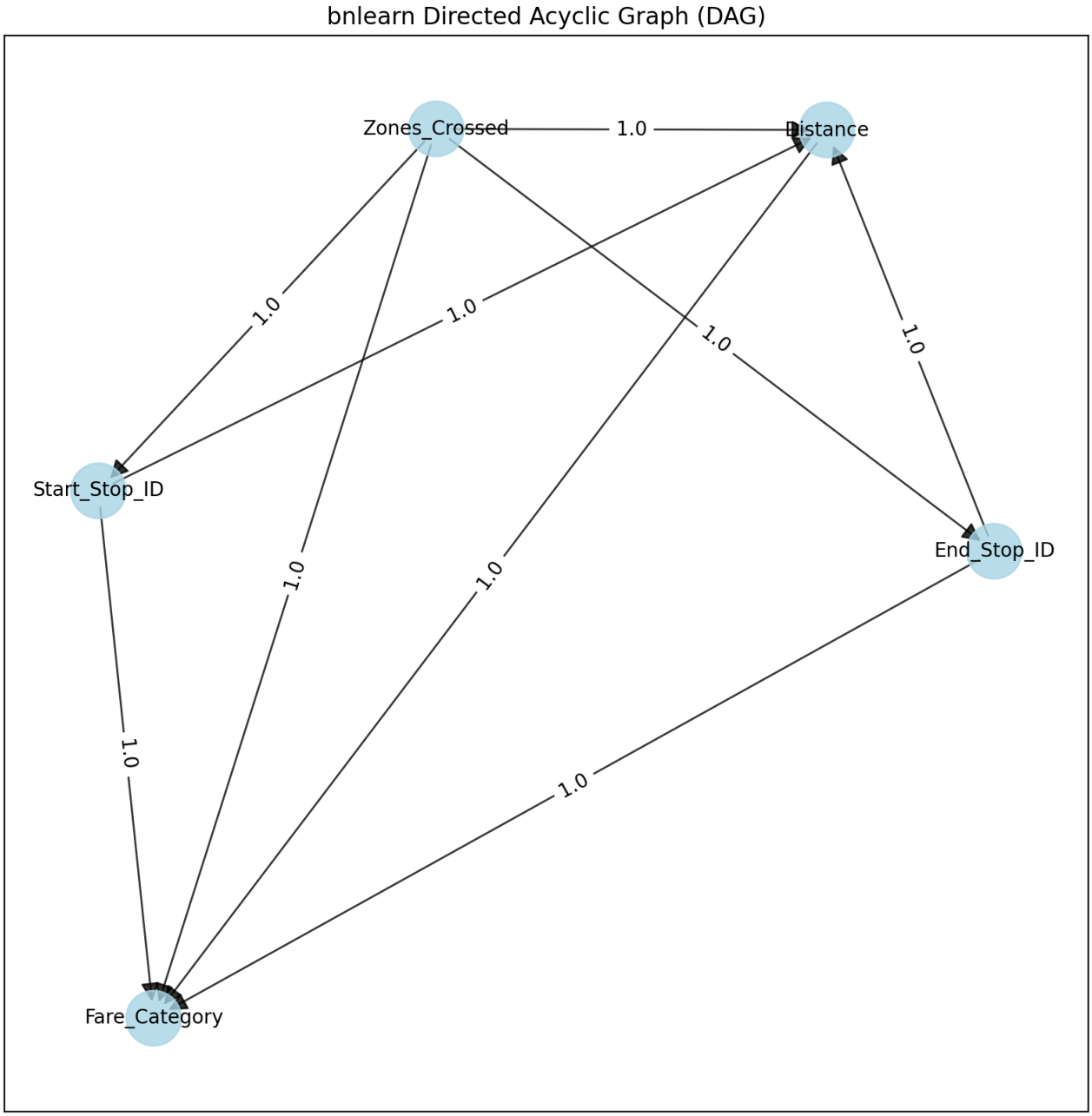
3. **Naive Bayes**: The Naive Bayes algorithm was used to optimize the Bayesian network. Naive Bayes is a special case of Bayesian Model where the only edges in the model are from the feature variables to the dependent variable.




Time taken using Naive Bayes Optimization:

```
Optimized Model (Naive Bayes): Time = 7.82s, Memory = 554.26KB
```

4. **Constraint-Based Search**: The constraint-based search algorithm was used to optimize the Bayesian network. The algorithm uses conditional independence tests to identify the structure of the Bayesian network.




Time taken using Constraint-Based Search Optimization:

```
Optimized Model (Constraint Based): Time = 2808.38s, Memory = 12398546.65KB
```

We find that even the optimization algorithim that performed the worst in terms of time efficiency i.e. Constraint-Based Search, still took less time than the initial Bayesian Network as well as the pruned Bayesian Network. This is because the number of edges has significantly reduced, and hence the number of computations has reduced.

Accuracy obtained for all the above optimization techniques was 100%.

```
+----+-----------------+-------------+
Total Test Cases: 350
Total Correct Predictions: 350 out of 350
Model accuracy on filtered test cases: 100.00%
```

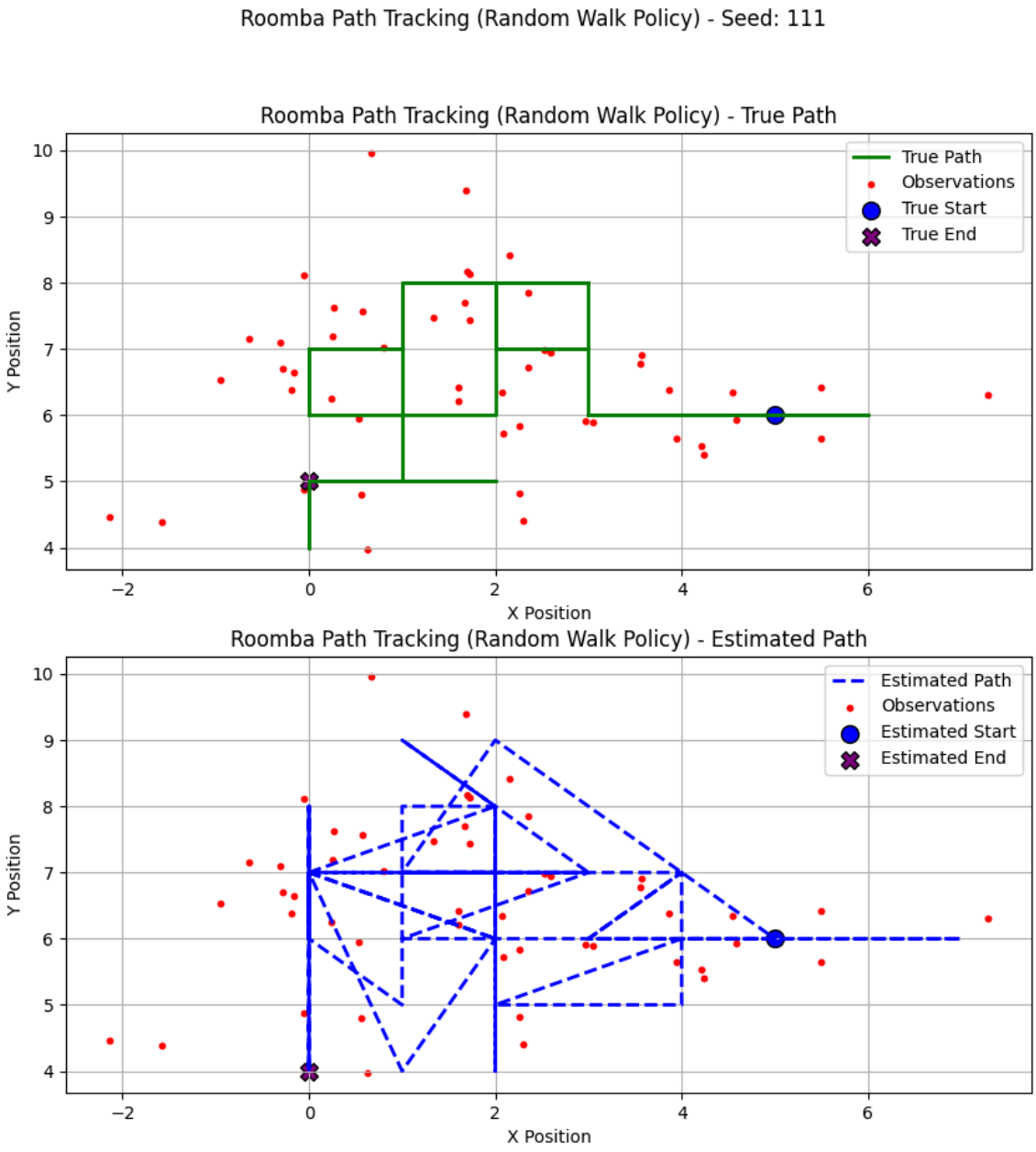
## Question 2

In this question the seed values selected were 111, 222, 333, 444, 555. We repeatedly achieved a higher accuracy for the Straight until Obstacle Roomba Policy when compared to the Random Walk Policy.

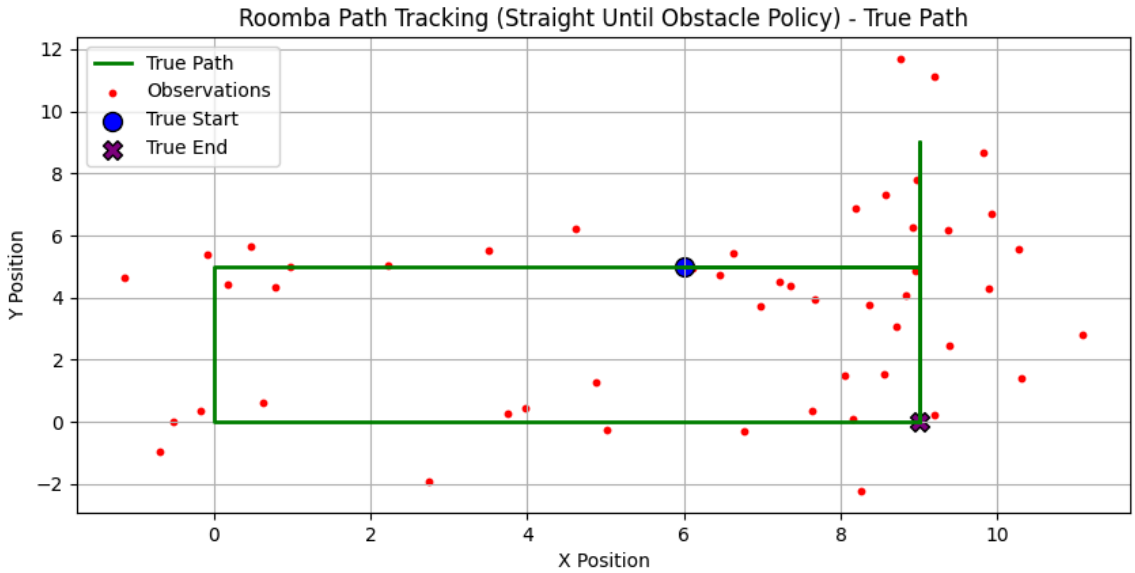
The maximum accuracy achieved for Straight until Obstacle Roomba Policy was 70% and for Random Walk Policy was 32%.

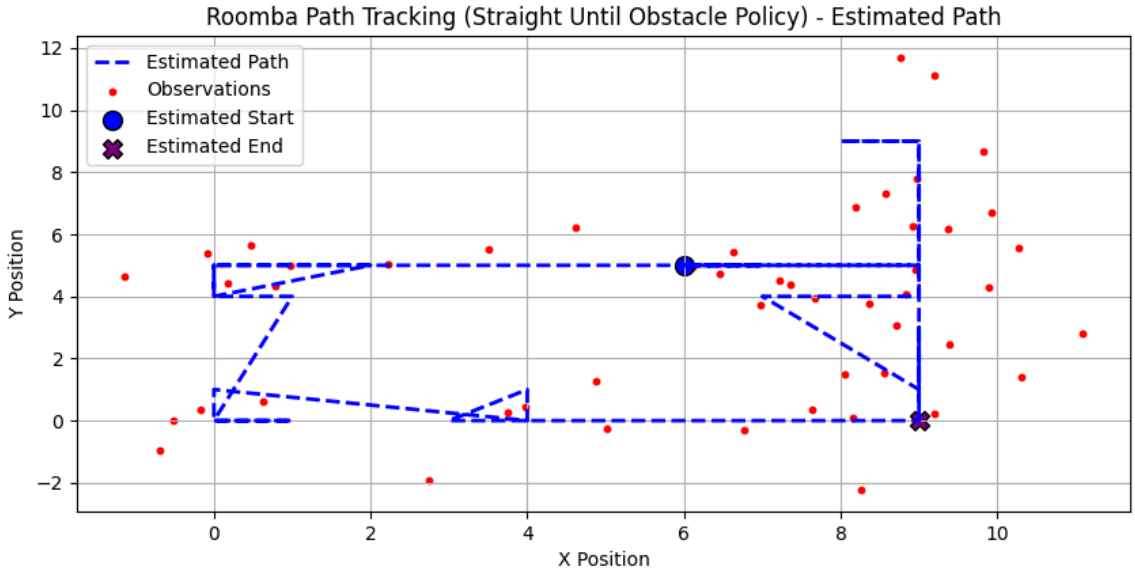
This is because the Straight until Obstacle Roomba Policy is more deterministic and hence has a higher accuracy. The Random Walk Policy is more random and hence has a lower accuracy.

The following graphs show the paths for both policies for the different seed values:

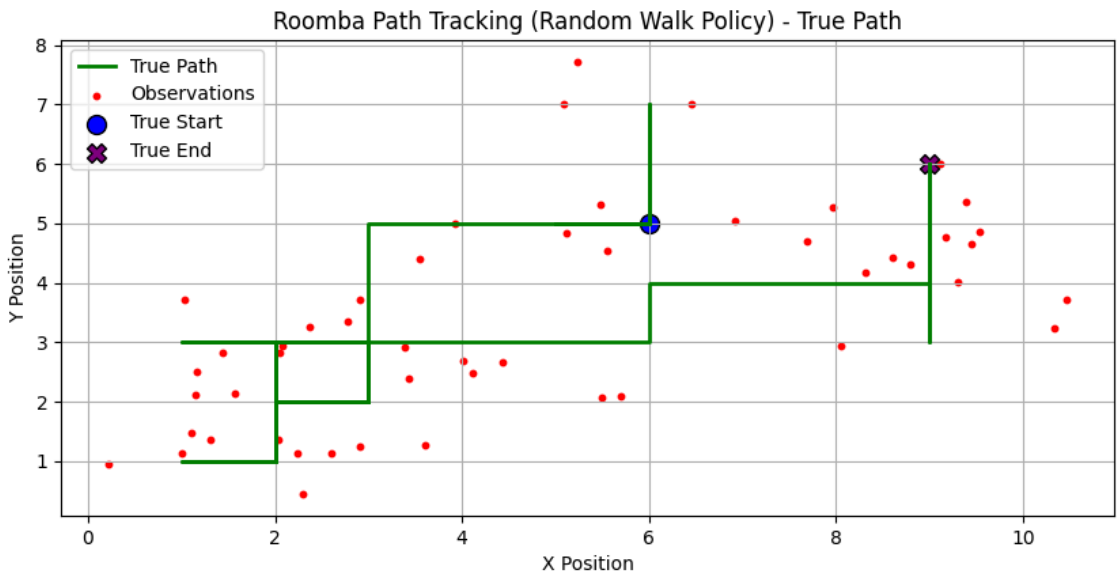


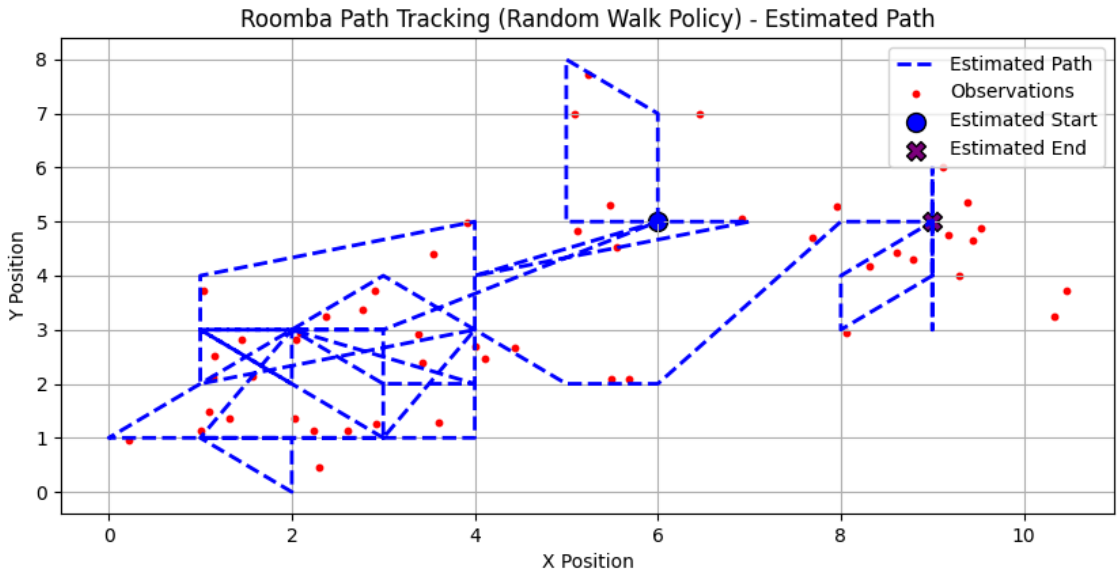
Roomba Path Tracking (Straight Until Obstacle Policy) - Seed: 111



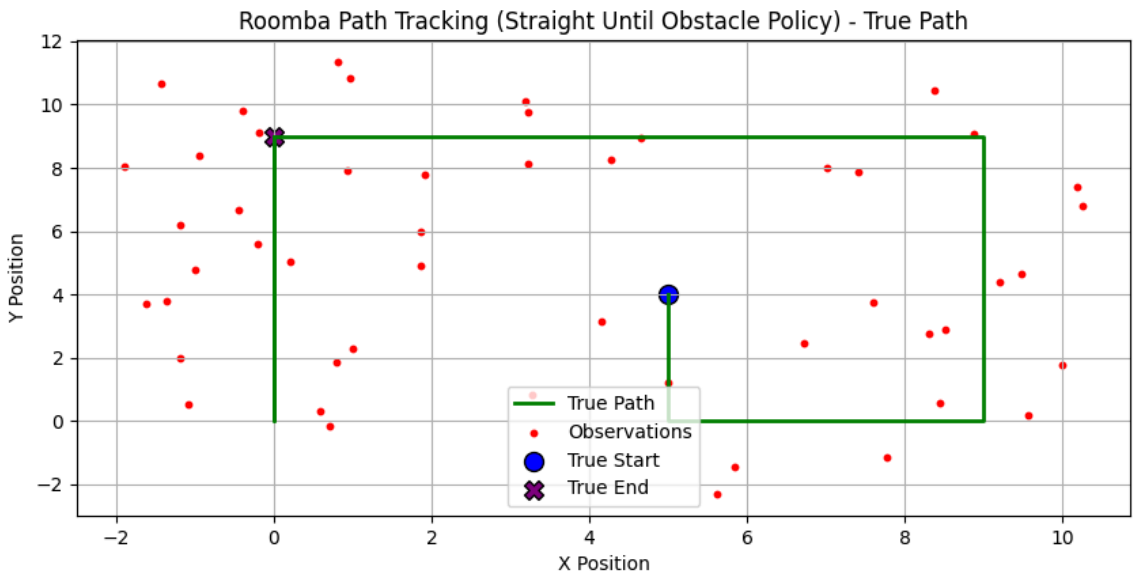



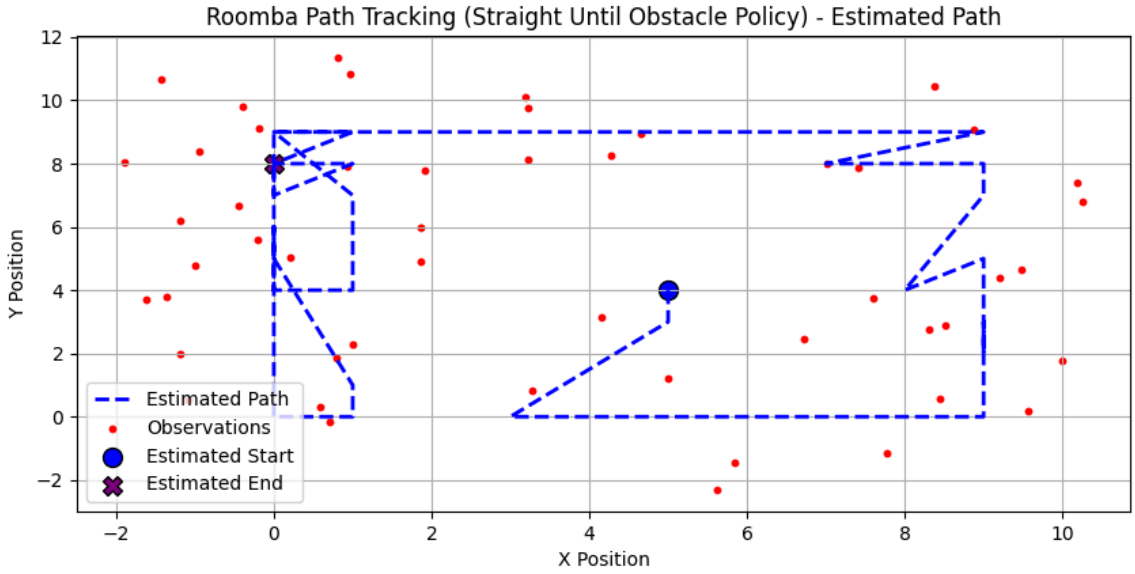
Roomba Path Tracking (Random Walk Policy) - Seed: 222



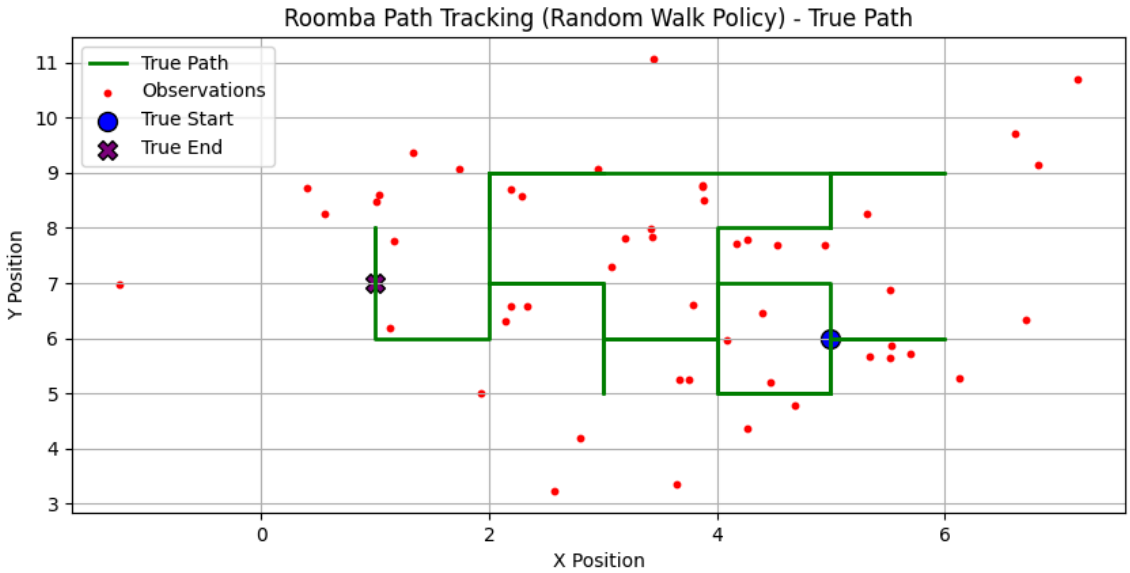



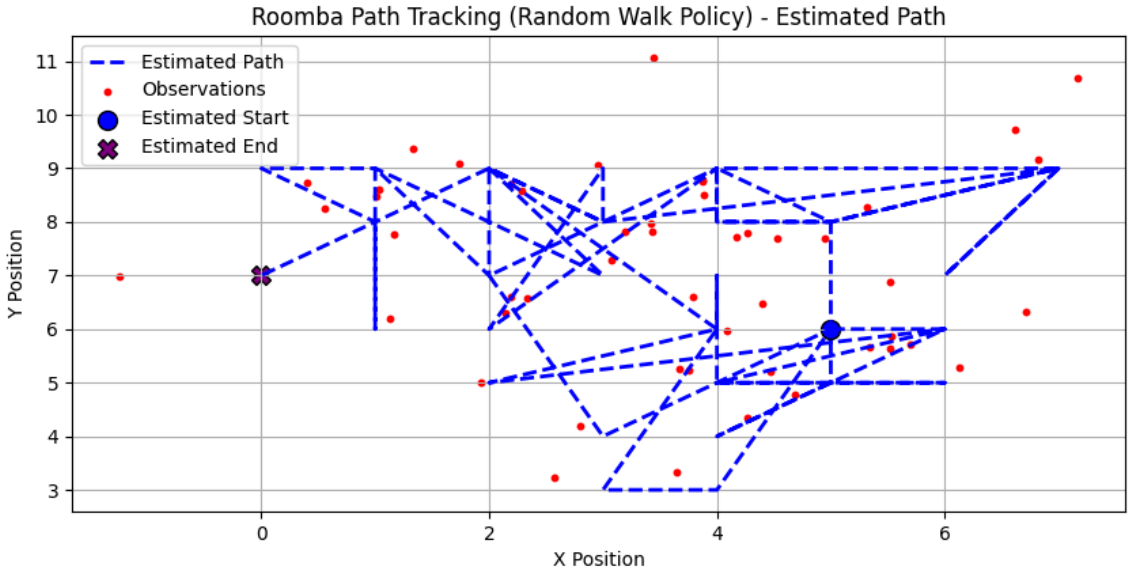
Roomba Path Tracking (Straight Until Obstacle Policy) - Seed: 222



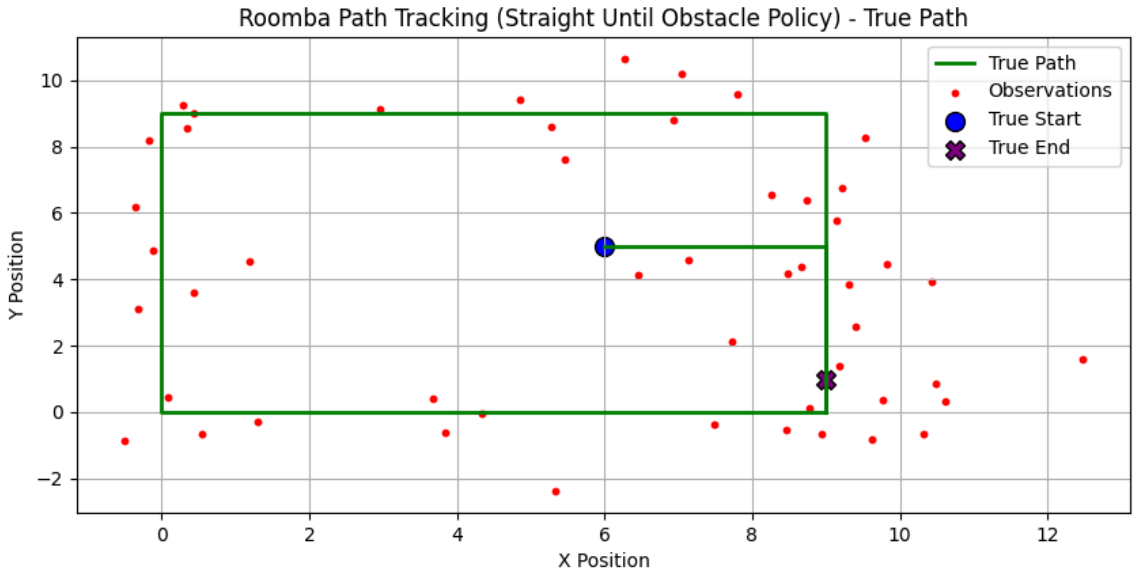



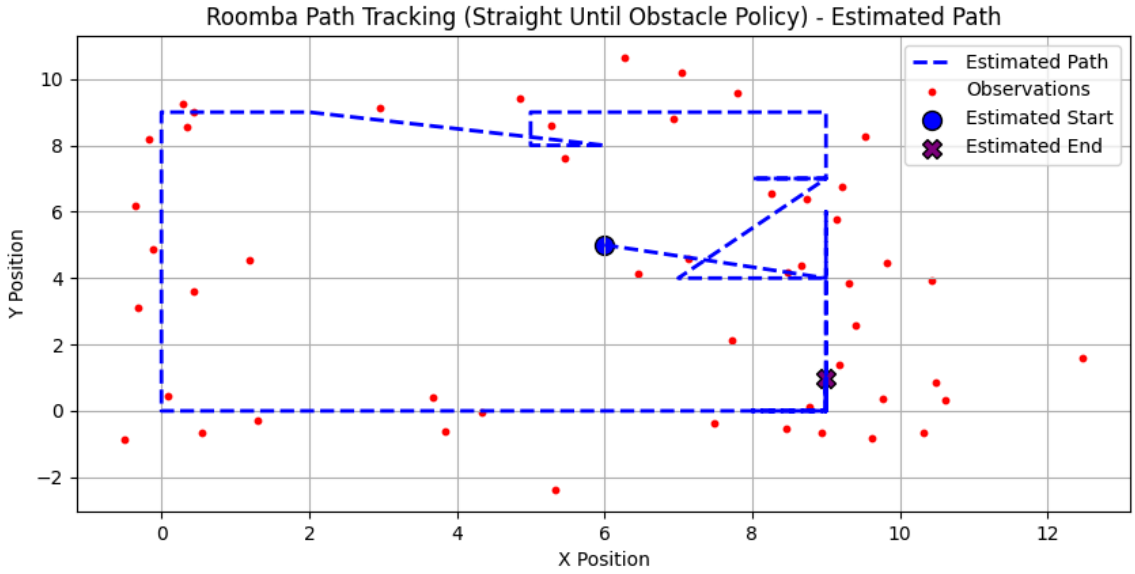
Roomba Path Tracking (Random Walk Policy) - Seed: 333



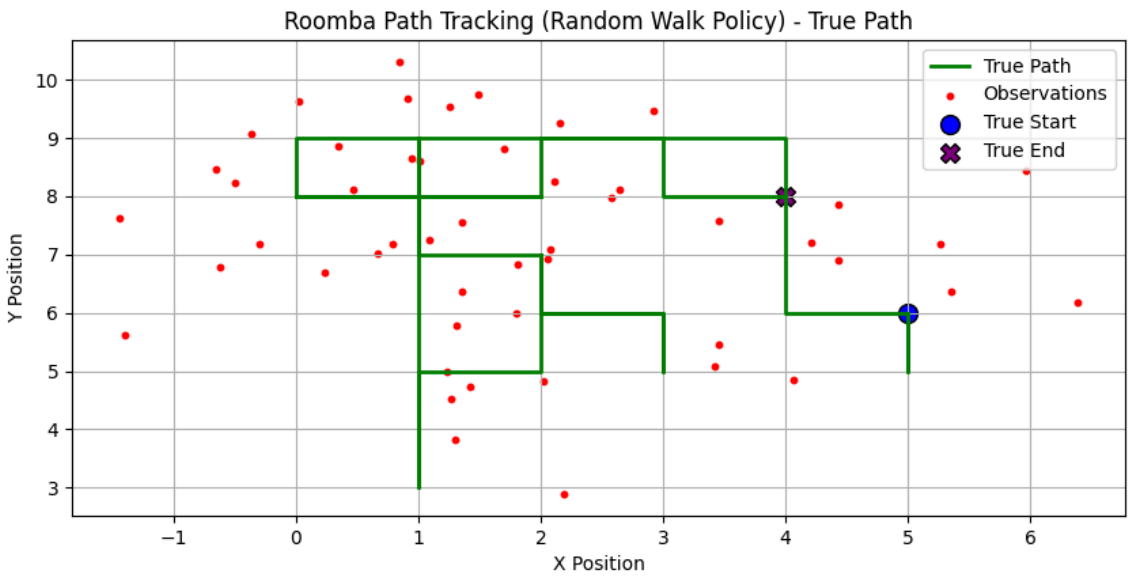



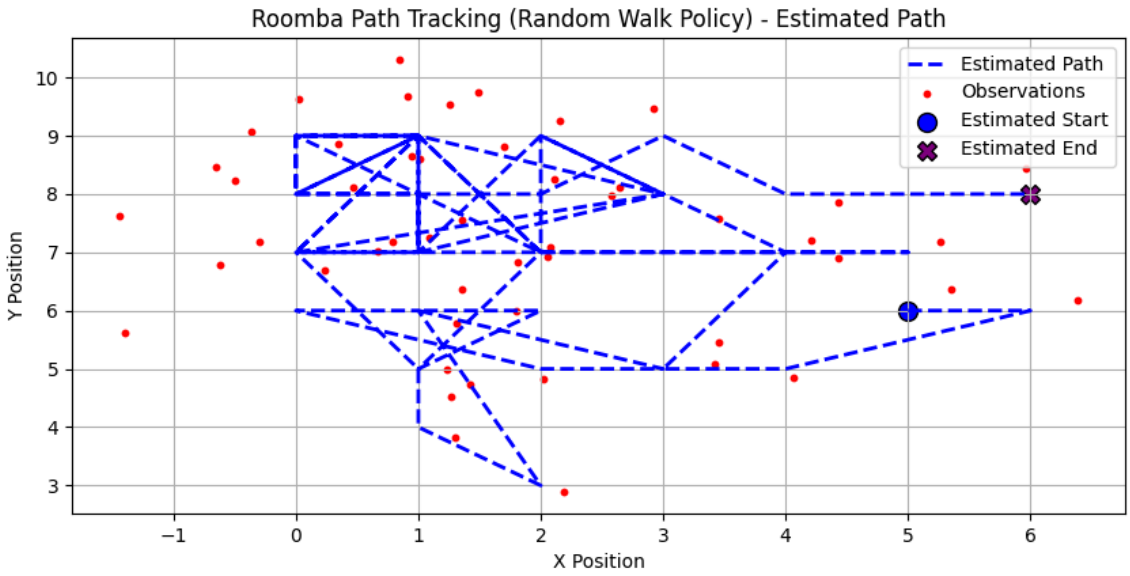
Roomba Path Tracking (Straight Until Obstacle Policy) - Seed: 333



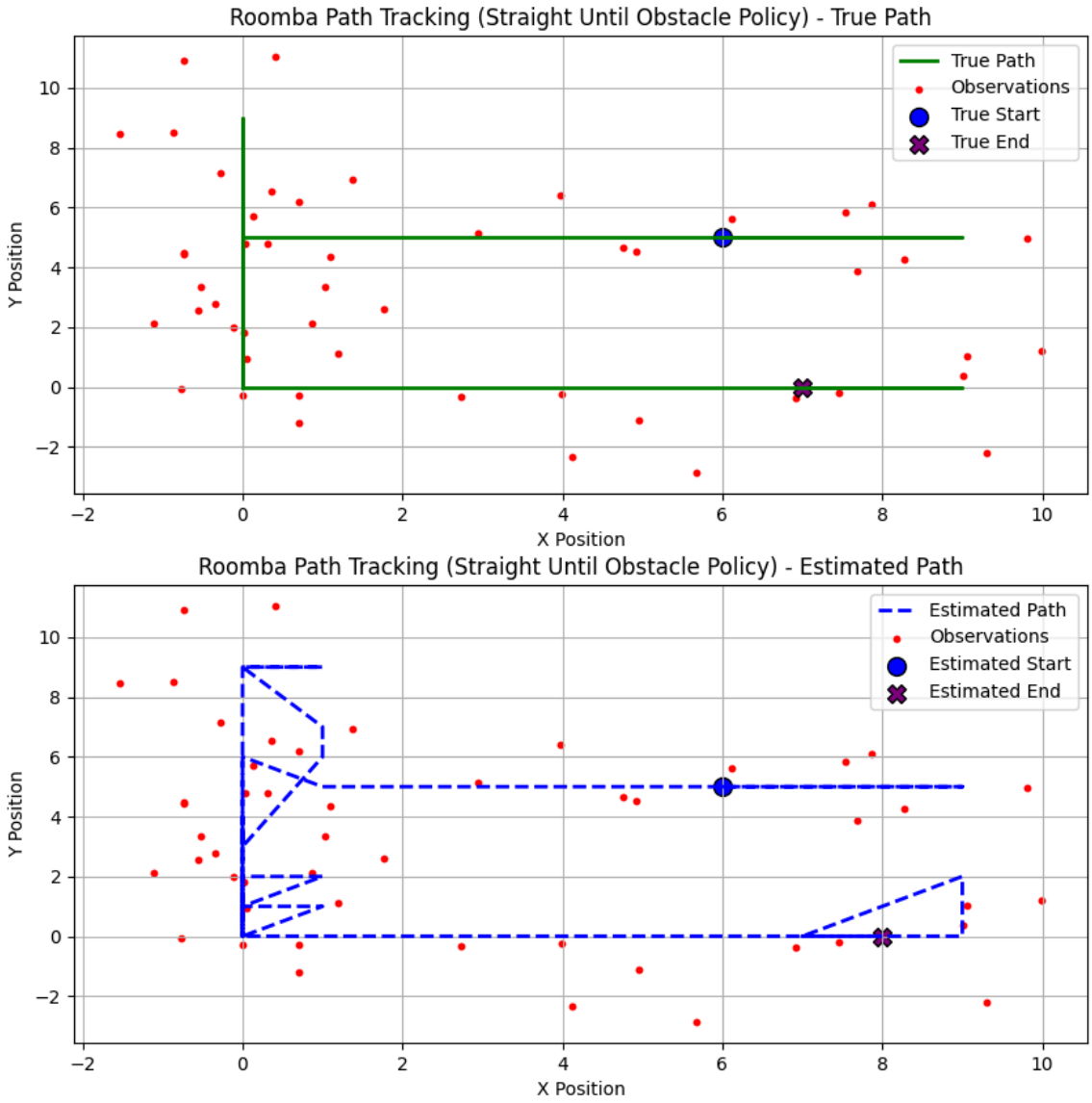



Roomba Path Tracking (Random Walk Policy) - Seed: 444

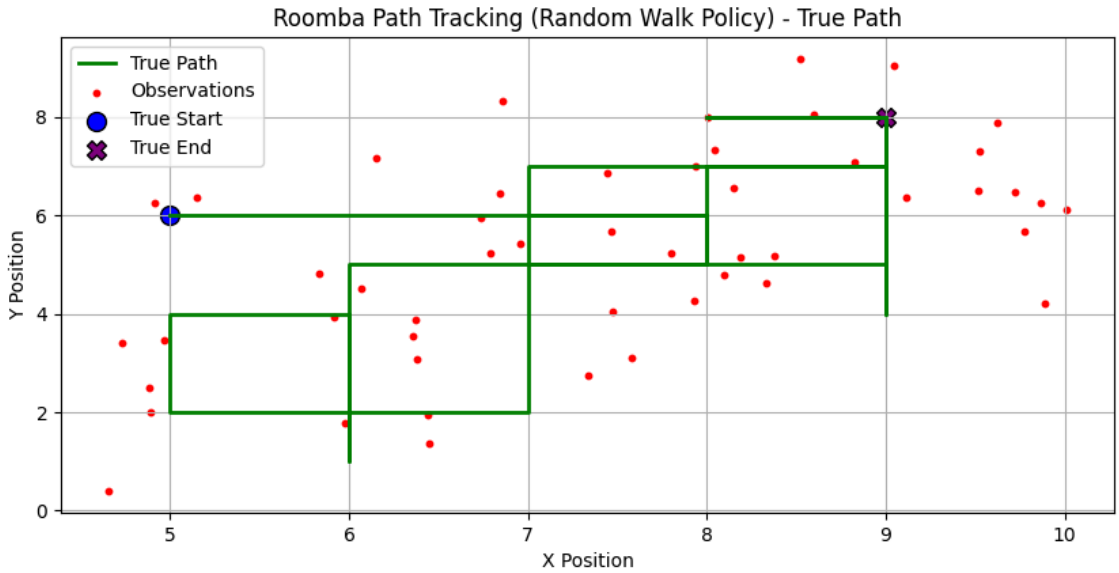


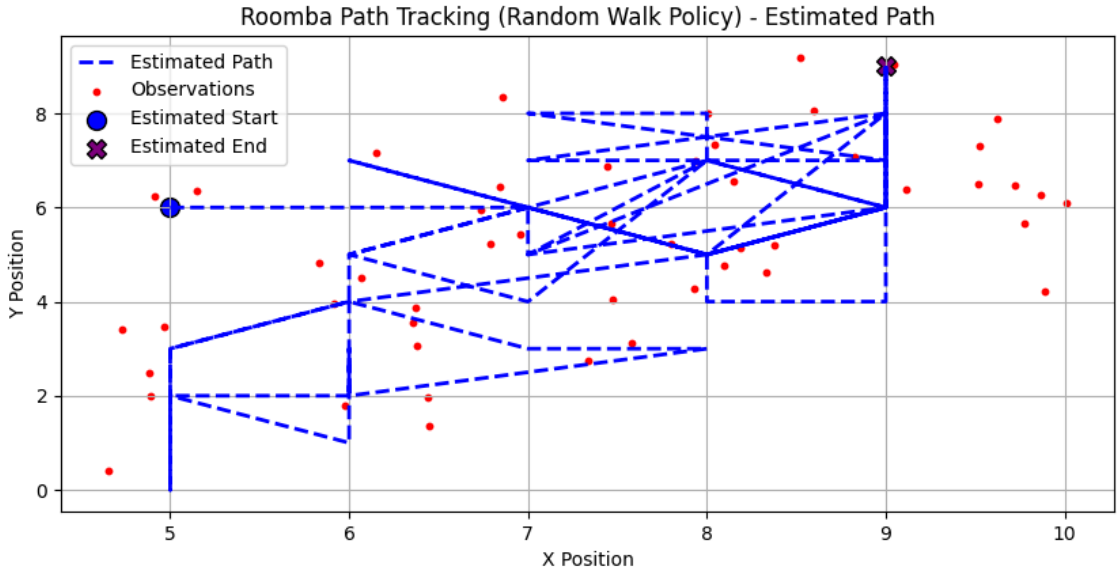



Roomba Path Tracking (Straight Until Obstacle Policy) - Seed: 444



Roomba Path Tracking (Random Walk Policy) - Seed: 555



Roomba Path Tracking (Straight Until Obstacle Policy) - Seed: 555