Technion – Israel Institute of Technology

HW4

Autonomous Navigation and Perception

086762

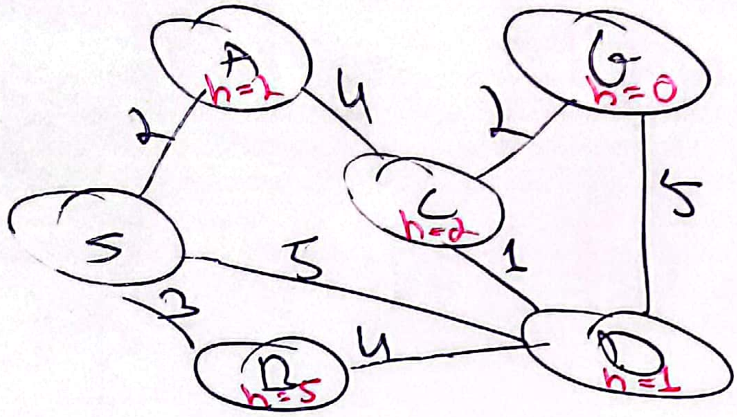
Alon Spinner	305184335	alonspinner@gmail.com
Dan Hazzan	308553601	danhazzan@campus.technion.ac.il

May 16, 2022

Theoretical Questions:

For this section, we decided to provide our answers in a hand-written format. The Hands-on section can be found right after the scanned pages.

1.



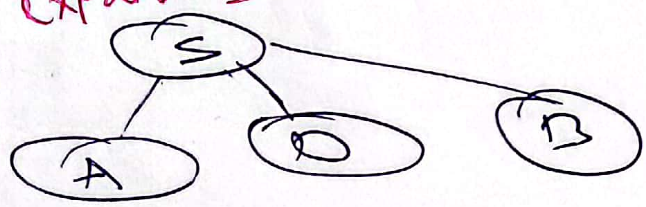
(a) BFS:

init

I. 

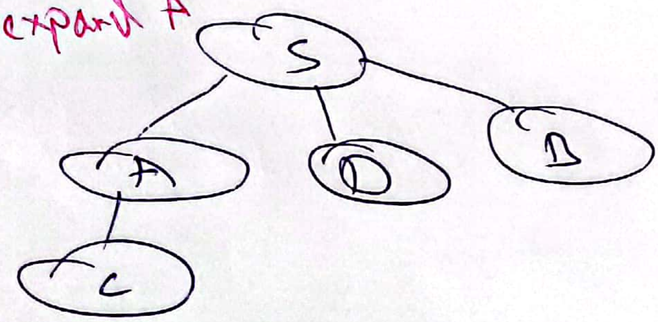
open = {S}

expand S

II. 

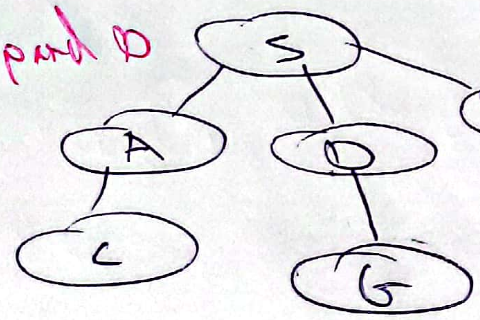
open = {A, D, B}

expand A

III 

open = {C, D, B}

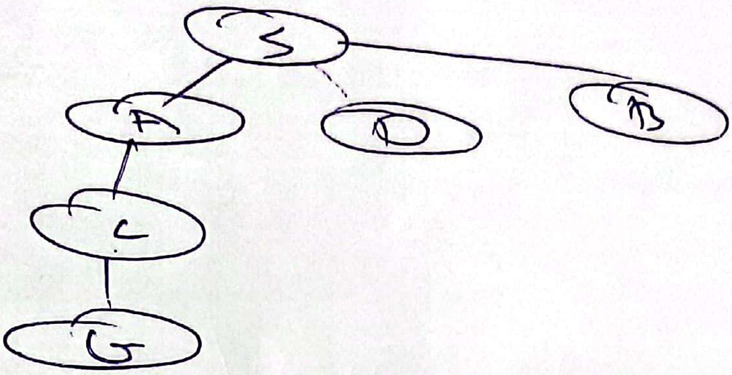
IV. expand D



open = {C, G, B}

IV expand C

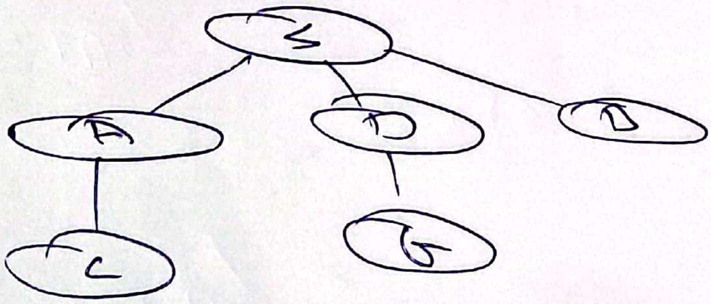
open = {D, B, G}



V expand G ⇒ open = {D, B}

VI expand D

open = {B, G}



VII expand G → open = {B}

VIII expand D ⇒ open = {}

shortest path: S → D → G

$d_G = 2$

V. expand B ⇒ open = {C, G}

VI. expand C → open = {G}

VII. expand G ⇒ open = {}

shortest path: S → D → G

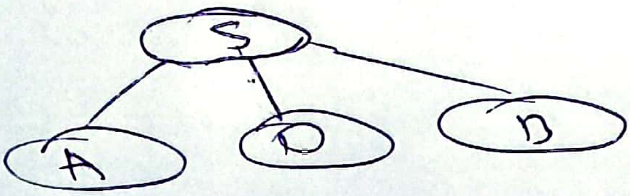
$d_G = 2$

(b) DFS:

I. init

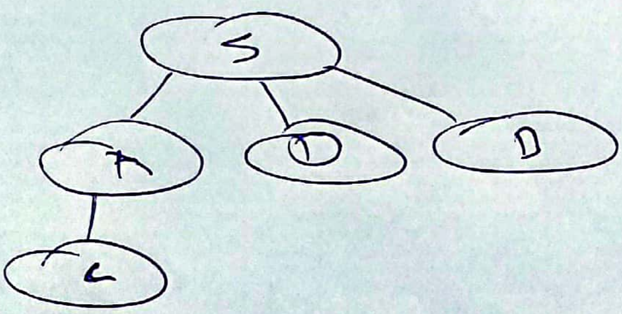
open = {S}

II expand S



open = {A, D, B}

III expand A



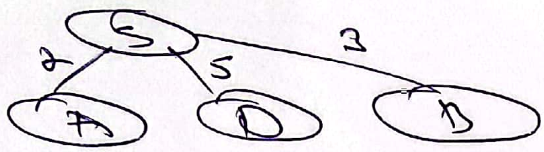
open = {C, D, B}

(c) Dijakstra:

I. init

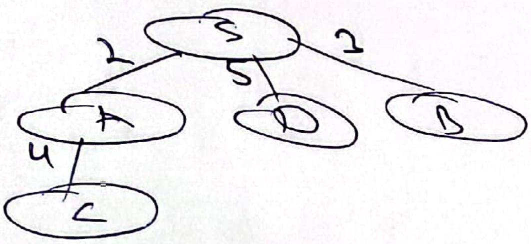
open = {S}

II. expand S



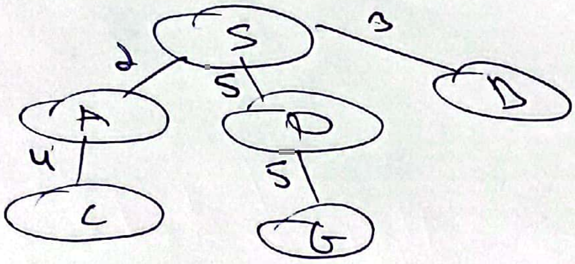
open = {A, D, B}

III. expand A



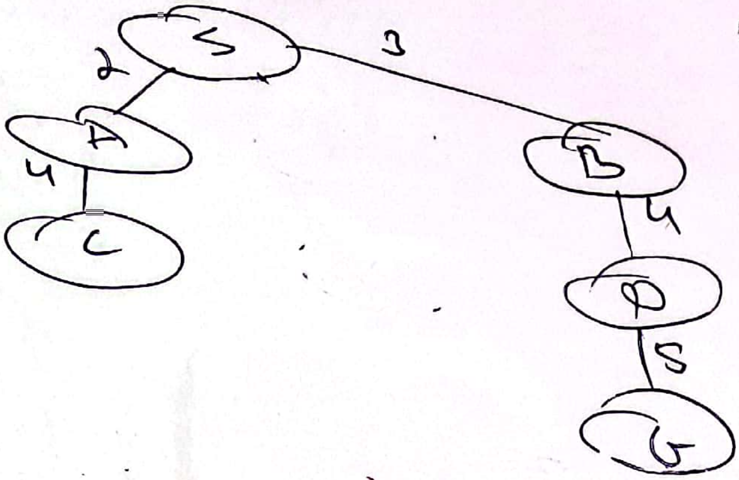
open = {C, D, B}

IV expand D



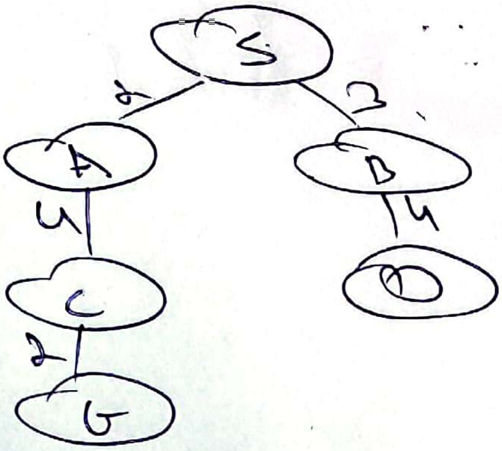
open = {C, G, B}

I expand B



open = {C, G, D}

V expand C



open = {G, D}

VI expand D ⟹ open = {G}

VI expand G ⟹ open = {}

shortest path

S → A → C → G

$g_G = 5$

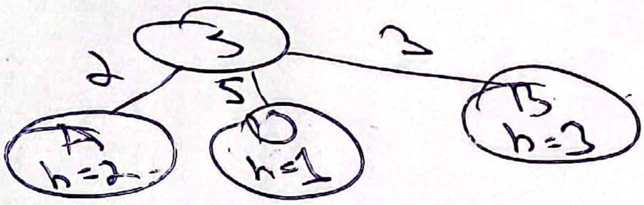
(d) A* :

I. init

S

open = {S}

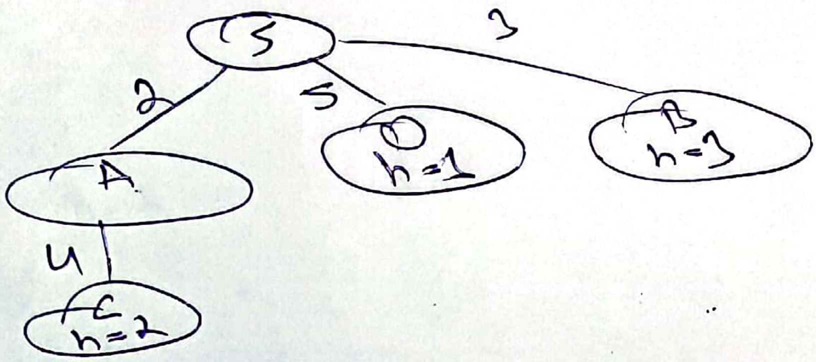
II. expand S



open = {A, D, B}

closed = {S}

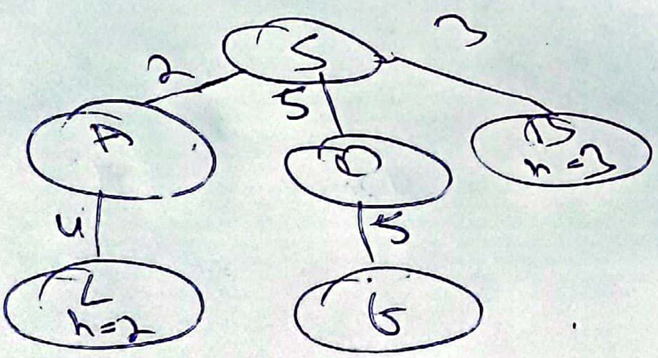
III. expand A



open = {D, B, C}

closed = {S, A}

IV. expand D



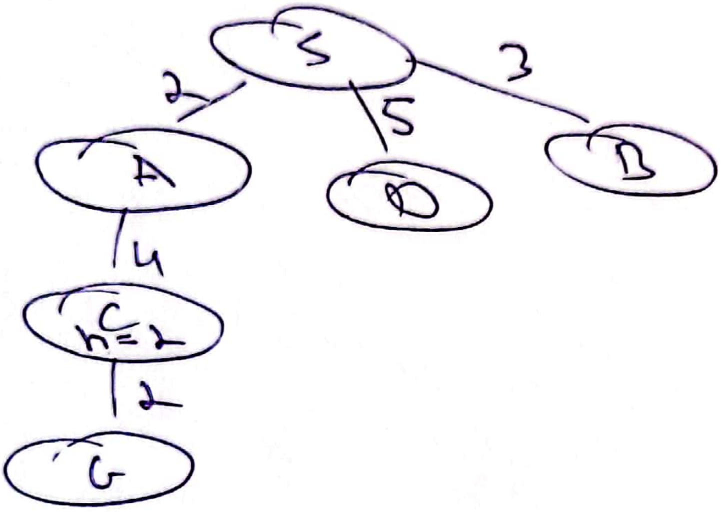
open = {B, C}

closed = {S, A, D}

V expand B ⟹ open = {C}

closed = {S, A, D, B}

VI expand C



open = {G}
closed = {S, A, D, B, C}

VII expand G ⟹ terminate

shortest path: S → A → C → G

$d_G = 8$

e. A^* was actually less efficient as it required 1 extra step which is due to a bad heuristic:

$h(C) = 2 \quad w(G, C) = 2$

$h(D) = 1 \quad w(G, D) = 5$

2.

(a) $h^{(1)}, h^{(2)}$ are consistent

show that $h = \max(h^{(1)}, h^{(2)})$ is also consistent

- $h(T) = \max(\underbrace{h^{(1)}(T)}_{0}, \underbrace{h^{(2)}(T)}_{0}) = 0$

- $c(x_i, x_j) + h(x_j) =$

$= c(x_i, x_j) + \max(h^{(1)}(x_j), h^{(2)}(x_j)) =$

$\left\{ \text{assume } h^{(1)}(x_j) \geq h^{(2)}(x_j) \text{ without loss of generality} \right\}$

$= c(x_i, x_j) + h^{(1)}(x_j) \geq c(x_i, x_j) + h^{(2)}(x_j) \geq$

$\geq h^{(2)}(x_i)$

consistency of $h^{(2)}$

also: $c(x_i, x_j) + h^{(1)}(x_j) \geq h^{(1)}(x_i)$

consistency of $h^{(1)}$

<u>hence:</u>

$$c(x_i, x_j) + h(x_j) \geq h^{(1)}(x_i)$$

$$c(x_i, x_j) + h(x_j) \geq h^{(2)}(x_j)$$

$$\Rightarrow c(x_i, x_j) + h(x_j) \geq \max\left(h^{(1)}(x_i), h^{(2)}(x_i)\right) =$$

$$= h(x_i)$$

(b) $h^{(1)}, h^{(2)}$ are consistent.

Show that $h = h^{(1)} + h^{(2)}$ is ε-consistent

- $h(T) = h^{(1)}(T) + h^{(2)}(T) = 0$

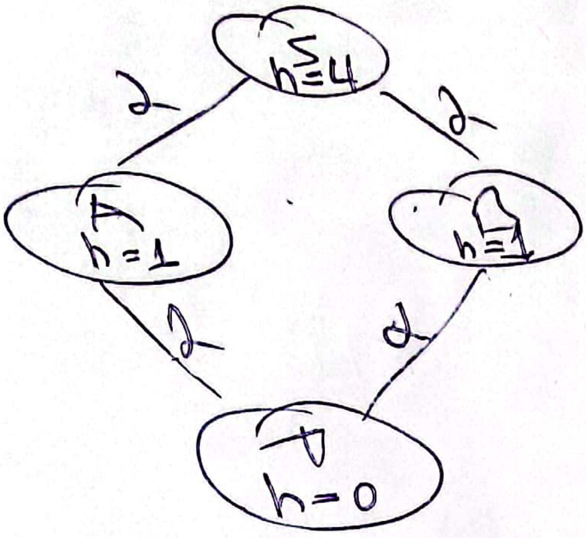
- $(+)\begin{cases} h^{(1)}(x_j) + c(x_i, x_j) \geq h^{(1)}(x_i) \\ h^{(2)}(x_j) + c(x_i, x_j) \geq h^{(2)}(x_i) \end{cases}$

$$h^{(1)}(x_j) + h^{(2)}(x_j) + 2c(x_i, x_j) \geq h^{(1)}(x_i) + h^{(2)}(x_i)$$

$$h(x_j) + 2c(x_i, x_j) \geq h(x_i)$$

$\Rightarrow \varepsilon = 2$ consistent

(c) example 2



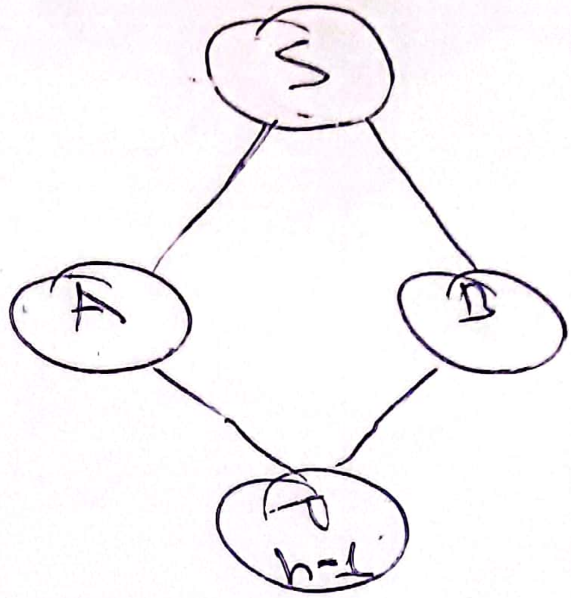
$h(S) \ ? \ h(B) + w(S,B)$

$4 > 3$

$\Downarrow$

$h(S) \not\leq h(B) + w(S,B)$

example 1



$h(T) = 1$

Hands On:

In the first question (a) we are tasked to create maps, and in the second question (b) solve a shortest path problem via graph forward search.
We decided to provide the plots for both questions together.

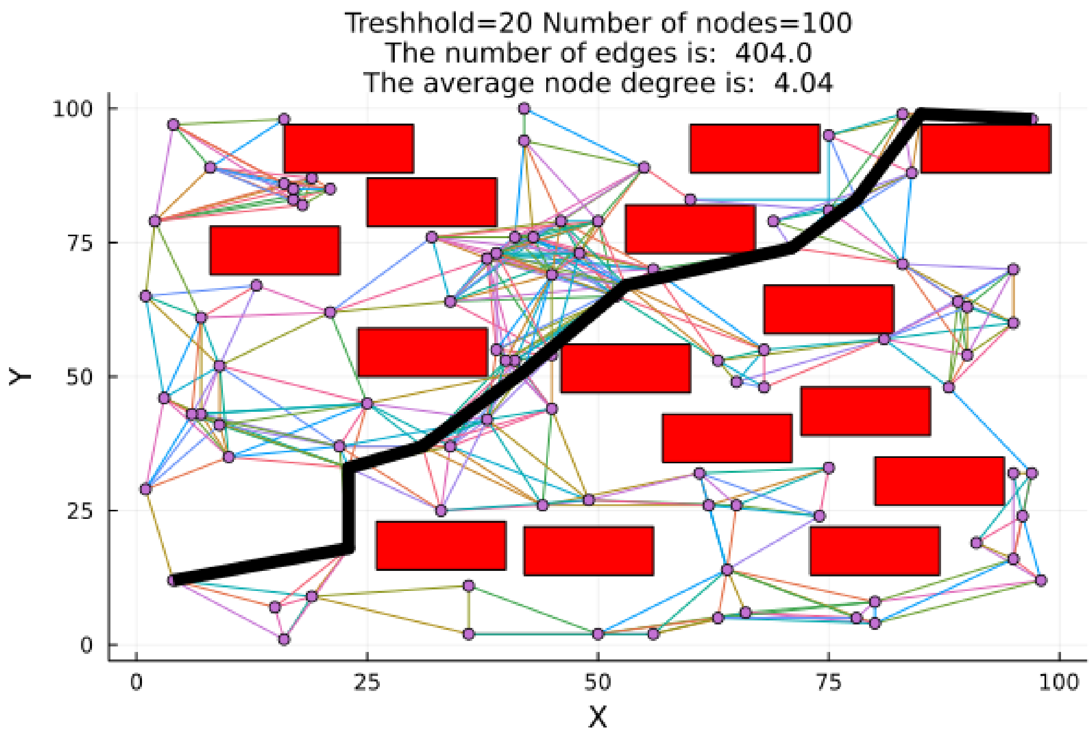
We built a map and a PRM with the following setting as follows:

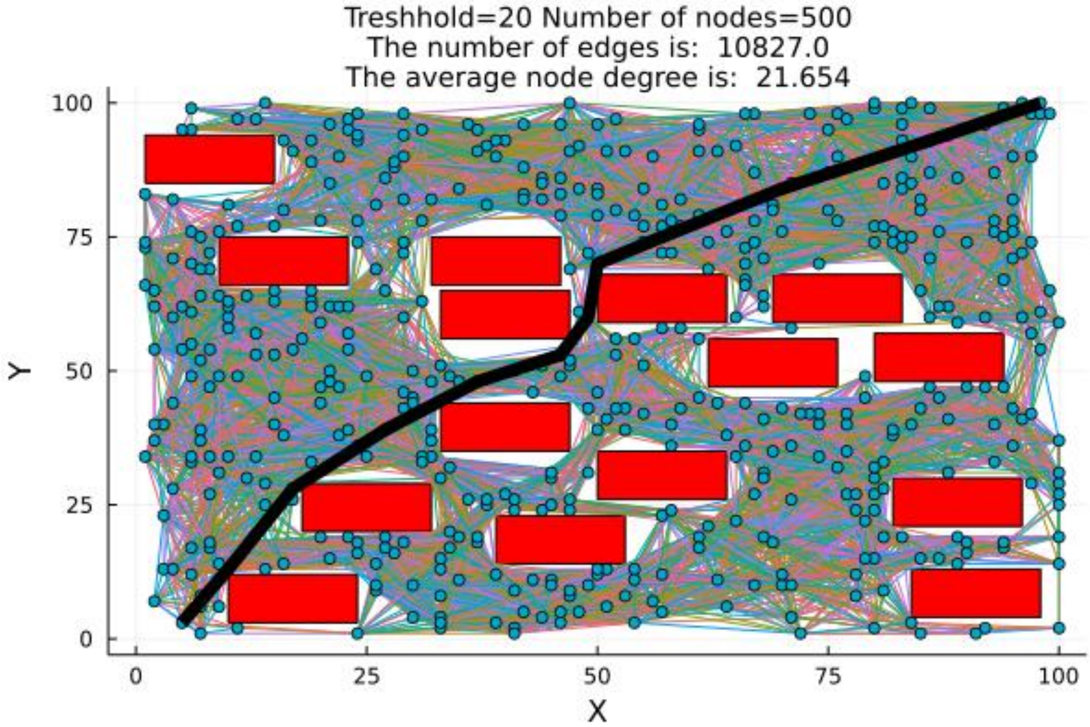
- Create a map where $(x, y) \in [0,100] \times [0,100]$
- Discretize the map to unit intervals, and place 15 rectangles of size 15×10 such that they don't overlap. Overlapping is checked by grid's occupancy
- Build PRM nodes with N_{nodes} = 100 or 500 by uniform sampling of the space. If a node falls inside a rectangle, it is discarded, and a new node is sampled instead.
- Connect the PRM nodes with edges such that each node is connected to all nodes whose distance to it is smaller than $th_d \in \{20,50\}$ and where the edge doesn't cross a rectangle. We checked for crossings via line intersection (each rectangle has 4 lines to check with).

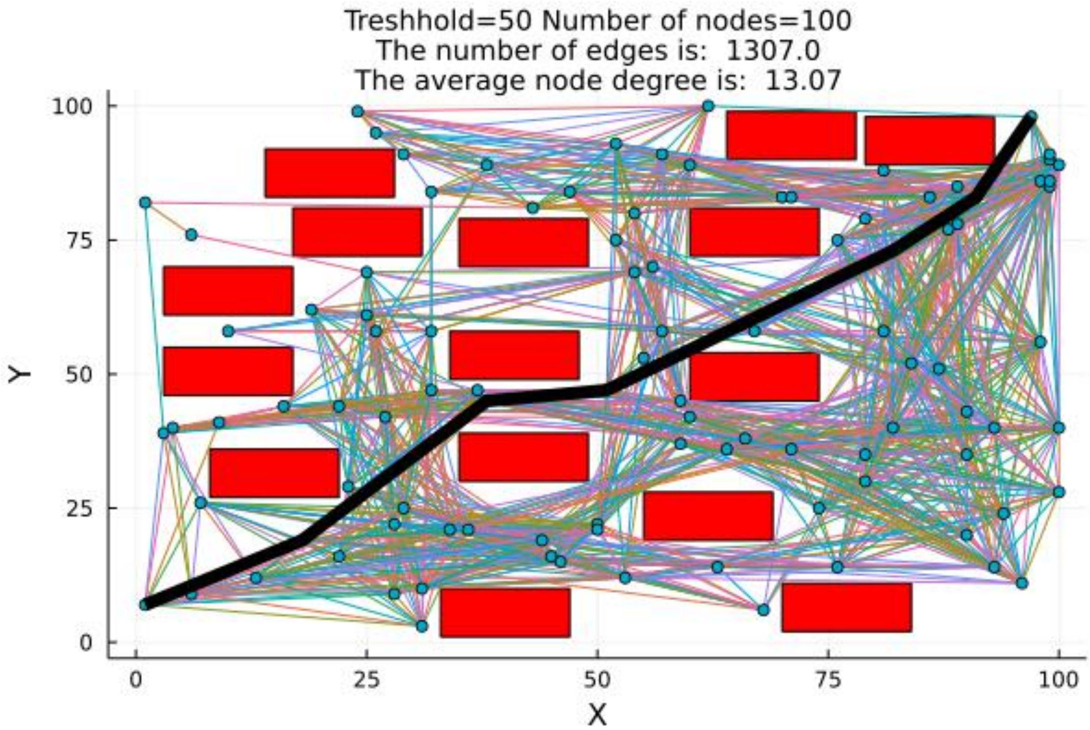
We also implemented both Dijkstra and A^* algorithms for *"SimpleWeightedGraph.jl"* in *"graphForwardSearch.jl"*

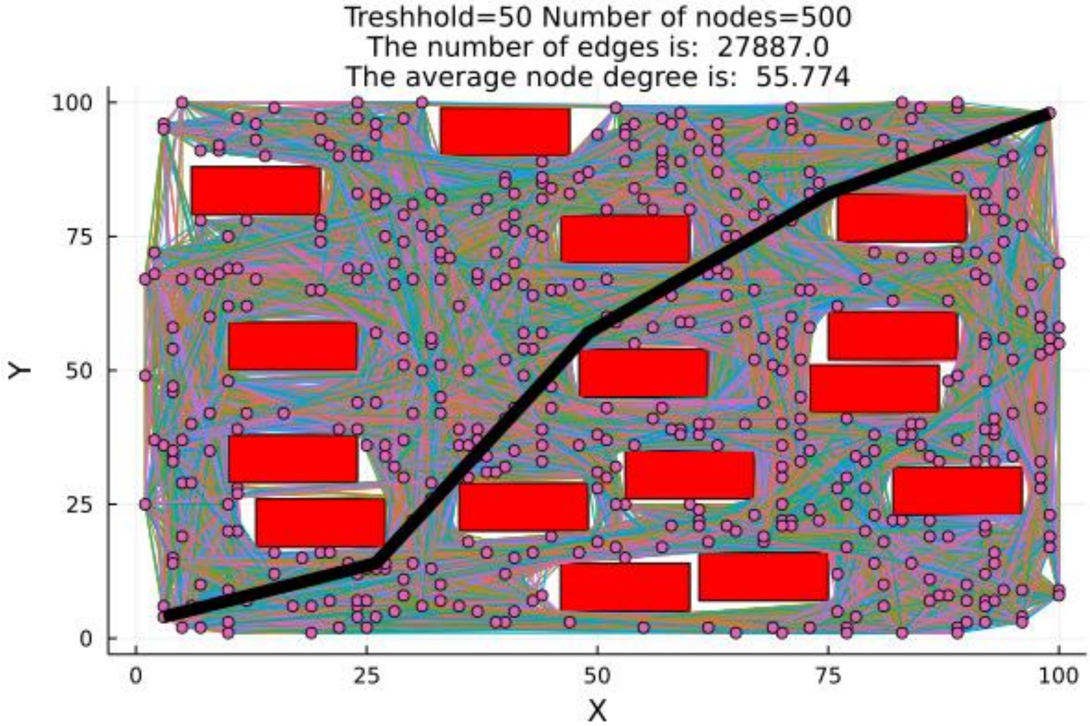
In *"Q1 hands on.jl"* we build the map, and searched for the shortest path between bottom left node to top right node (chosen by distance from [0,0]) on the PRM via A^* with Euclidian distance to goal heuristics.

We show below the outcomes for all combinations N_{nodes}, th_d with information of number of edges and average node degree.




Treshhold=20 Number of nodes=500
The number of edges is: 10827.0
The average node degree is: 21.654
Y
X
0
25
50
75
100


Treshhold=50 Number of nodes=100
The number of edges is: 1307.0
The average node degree is: 13.07
Y
X
0
25
50
75
100



Treshhold=50 Number of nodes=500
The number of edges is: 27887.0
The average node degree is: 55.774
100
75
50
25
0
Y
0
25
50
75
100
X