CSIT 6000F

Assignment 1

Yuchen, GUO No.20477118

September 20, 2017

1 Problem 1

In this problems, we can describe the input of the sensors using a matrix:

$$S = \begin{bmatrix} s_1 & s_2 & s_3 \\ s_8 & - & s_4 \\ s_7 & s_6 & s_5 \end{bmatrix}$$

So the robot has only 9 exclusive states as following:

$$S_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & - & 0 \\ 1 & 0 & 0 \end{bmatrix}, S_2 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & - & 0 \\ 0 & 0 & 0 \end{bmatrix}, S_3 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & - & 1 \\ 0 & 0 & 1 \end{bmatrix},$$
$$S_4 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & - & 0 \\ 1 & 0 & 0 \end{bmatrix}, S_5 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & - & 0 \\ 0 & 0 & 0 \end{bmatrix}, S_6 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & - & 1 \\ 0 & 0 & 1 \end{bmatrix},$$
$$S_7 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & - & 0 \\ 1 & 1 & 1 \end{bmatrix}, S_8 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & - & 0 \\ 1 & 1 & 1 \end{bmatrix}, S_9 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & - & 1 \\ 1 & 1 & 1 \end{bmatrix},$$

The state of every cell is shown in Figure 1.1.

S1	S2	S2	S2	S2	S2	S2	S2	S2	S3
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S4	S5	S5	S5	S5	S5	S5	S5	S5	S6
S7	S8	S8	S8	S8	S8	S8	S8	S8	S9

Figure 1.1: State of every cell in the grid.

Below are the boolean forms of S_1 to S_9:

$$
\begin{aligned}
S_1 &= s_1 \cdot s_2 \cdot s_3 \cdot \overline{s_4} \cdot \overline{s_5} \cdot \overline{s_6} \cdot s_7 \cdot s_8 \\
S_2 &= s_1 \cdot s_2 \cdot s_3 \cdot \overline{s_4} \cdot \overline{s_5} \cdot \overline{s_6} \cdot \overline{s_7} \cdot \overline{s_8} \\
S_3 &= s_1 \cdot s_2 \cdot s_3 \cdot s_4 \cdot s_5 \cdot \overline{s_6} \cdot \overline{s_7} \cdot \overline{s_8} \\
S_4 &= s_1 \cdot \overline{s_2} \cdot \overline{s_3} \cdot \overline{s_4} \cdot \overline{s_5} \cdot \overline{s_6} \cdot s_7 \cdot s_8 \\
S_5 &= \overline{s_1} \cdot \overline{s_2} \cdot \overline{s_3} \cdot \overline{s_4} \cdot \overline{s_5} \cdot \overline{s_6} \cdot \overline{s_7} \cdot \overline{s_8} \\
S_6 &= \overline{s_1} \cdot \overline{s_2} \cdot s_3 \cdot s_4 \cdot s_5 \cdot \overline{s_6} \cdot \overline{s_7} \cdot \overline{s_8} \\
S_7 &= s_1 \cdot \overline{s_2} \cdot \overline{s_3} \cdot \overline{s_4} \cdot s_5 \cdot s_6 \cdot s_7 \cdot s_8 \\
S_8 &= \overline{s_1} \cdot \overline{s_2} \cdot \overline{s_3} \cdot \overline{s_4} \cdot s_5 \cdot s_6 \cdot s_7 \cdot \overline{s_8} \\
S_9 &= \overline{s_1} \cdot \overline{s_2} \cdot s_3 \cdot s_4 \cdot s_5 \cdot s_6 \cdot s_7 \cdot \overline{s_8}
\end{aligned}
$$

1.1

Let's take the northwest corner for example.

When the robot is not next to the north border, move it to north, otherwise move it to west

Here is the production system:

$$\overline{s_2} \longrightarrow north$$
$$\overline{s_8} \longrightarrow west$$

or

$$S_2 + S_3 \longrightarrow west$$
$$S_4 + S_5 + S_6 + S_7 + S_8 + S_9 \longrightarrow north$$

1.2

We can prove it by putting the robot in the center of the grid, and then prove it can't visit every cell. Here are my two methods:

1.2.1 Method One

With S_5, it can move in any direction in $north, west, east, south$, in Figure 1.2, we assume it be west, and it can be any direction by rotating the image, the situations are equivalent.
The robot will move in the same direction until it reaches a border. To visit all cells in the grid, it must visit other cells with $S_5 = 1$. So let's see how can its state transfer to S_5 again, all possible ways are marked with pointers in the left part of Figure 1.2. So in every path it enters a S_5 cell, it will go straight to the west border and can't visit other S_5 cells, so it's proved that no production system can make the robot visit all cells in the grid. The right part shows possible circles that the robot can move in(equivalent paths are ignored).

1.2.2 Method Two

Figure 1.3 shows all production systems makes the robot move between some cells(Not all cells in the grid).

The search tree expands in the following process:

We start with a S_5 cell, four actions are in fact equivalent as we rotate the grid, assume the production system gives $S_5 \rightarrow W$, it reaches the west border, the next decision is $S_4 \rightarrow ?$.

If $S_4 \rightarrow E$, the robot will move between two nodes.
So we just need to expand $S_4 \rightarrow N$ and $S_4 \rightarrow S$, and the two branches are equivalent too. Let's assume $S_4 \rightarrow N$.

Then the robot reaches the northwest corner: S_1,
If $S_1 \rightarrow S$, the robot will move between two nodes.

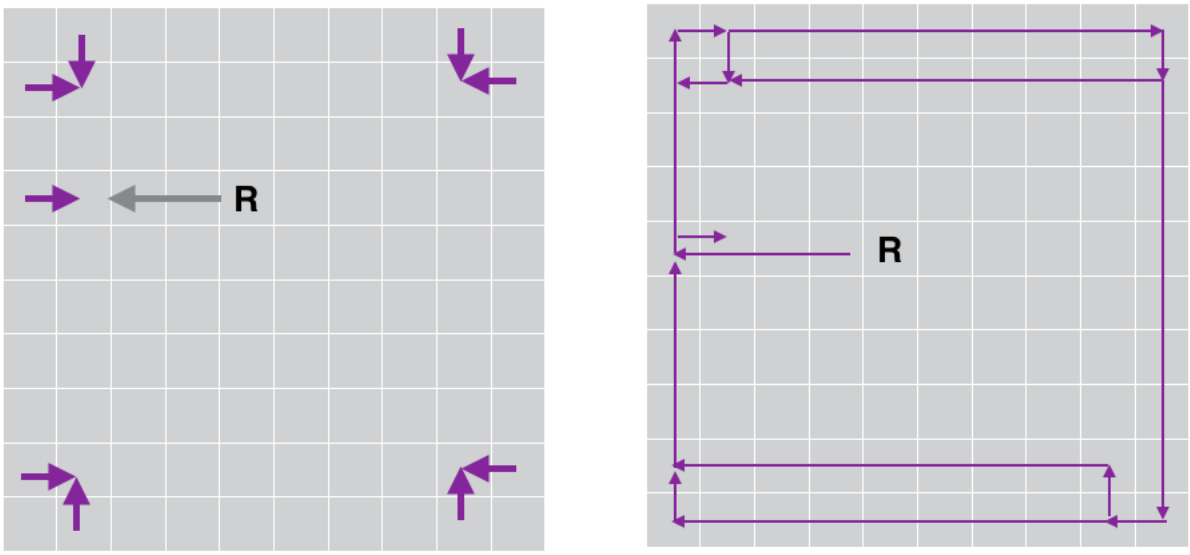

Figure 1.2: Ways to enter S_5 again and possible circles

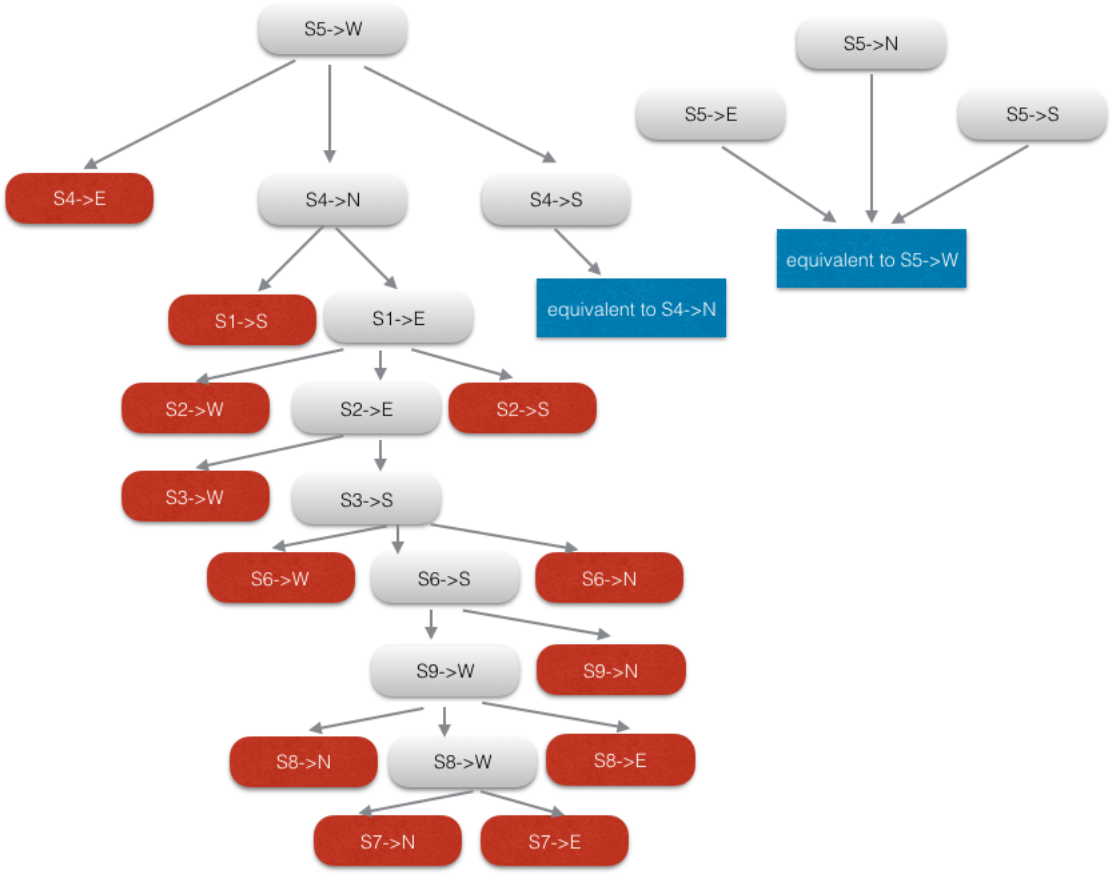

Figure 1.3: Search tree to find production system that can visit all cells, red nodes means the robot is going to move in a circle of several nodes, blue nodes means the situation is equivalent to another one, as we can see, all paths ends with a red node.

So $S_1 \rightarrow E$.

Then the state of the robot becomes S_2,
If $S_2 \rightarrow W$, the robot will move between two nodes.
If $S_2 \rightarrow S$, the robot will move in a circle of four nodes.
So $S_2 \rightarrow E$.

Then the state of the robot becomes S_3,
If $S_3 \rightarrow W$, the robot will move between two nodes.
So $S_3 \rightarrow S$.

Then the state of the robot becomes S_6,
If $S_6 \rightarrow N$, the robot will move between two nodes.
So $S_6 \rightarrow W$, the robot will move in a circle of twenty nodes.
So $S_6 \rightarrow S$.

Then the state of the robot becomes S_9,
If $S_9 \rightarrow N$, the robot will move between two nodes.
So $S_9 \rightarrow W$.

Then the state of the robot becomes S_8,
If $S_8 \rightarrow E$, the robot will move between two nodes.
So $S_8 \rightarrow N$, the robot will move in a circle of thirty-six nodes.
So $S_8 \rightarrow W$.

Then the state of the robot becomes S_7,
If $S_7 \rightarrow E$, the robot will move between two nodes.
So $S_7 \rightarrow N$.

So, the state of the robot becomes S_4 again, it will move clockwise next to the border, move in a circle of thirty-six nodes.

Thus it's proved that no production system can make the robot visit all cells in the grid.

1.3 Part 3 of Problem 1

Let W_{ij} be the features defined as:
W_{ij} = 1 iff at the previous time step, S_i = 1, and the robot moved j($j \in \{N, W, E, S\}$). For example: W_{1E} = 1 iff at the previous time step, S_1 = 1, and the robot moved $East$.

As Figure 1.4 shows, we want the robot to visit every cell in the route in the left picutre.

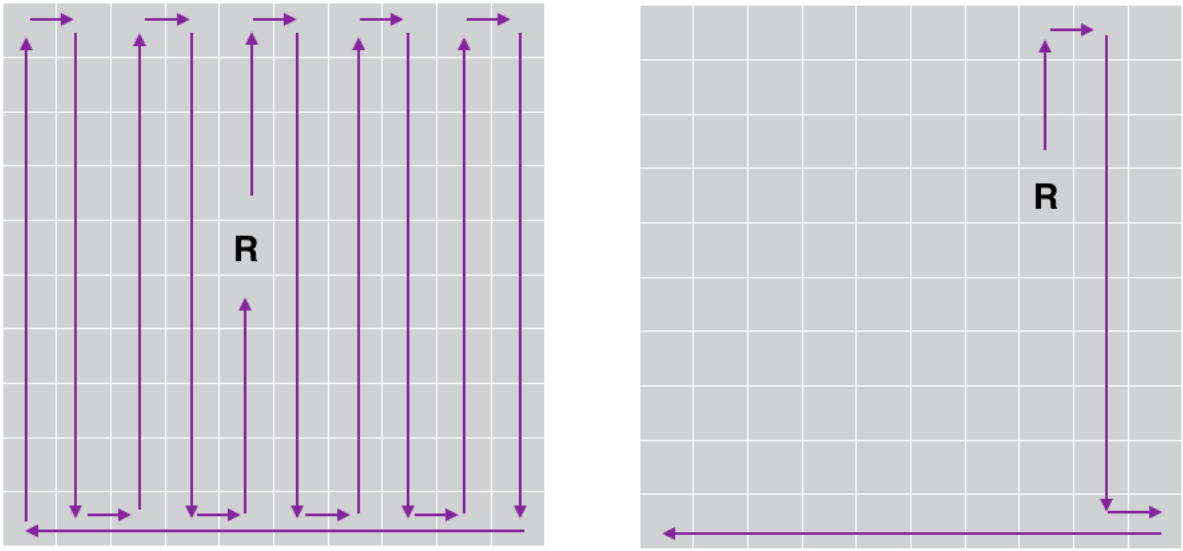

Figure 1.4: Two situations to visit all cells

Here is the production system:

$$S_2 \cdot W_{5N} \longrightarrow east$$
$$S_2 \cdot (W_{2E} + W_{1E}) \longrightarrow south$$
$$S_5 \cdot (W_{2S} + W_{5S}) \longrightarrow south$$
$$S_5 \cdot (W_{8N} + W_{5N}) \longrightarrow north$$
$$S_8 \cdot W_{5S} \longrightarrow east$$
$$S_8 \cdot W_{8E} \longrightarrow north$$
$$S_8 \cdot (W_{8W} + W_{9W}) \longrightarrow west$$
$$S_1 \longrightarrow east$$
$$S_3 + S_6 \longrightarrow south$$
$$S_4 + S_5 + S_7 \longrightarrow north$$
$$S_9 \longrightarrow west$$
$$1 \longrightarrow north$$

2 Problem 2

Let the input be A, B, C, D, E. Then:

$$\boldsymbol{Result} = (\mathbf{1.1} * \boldsymbol{A} + \mathbf{3.1} * \boldsymbol{B} - \boldsymbol{C} - \mathbf{2} * \boldsymbol{D} + \mathbf{0.5} * \boldsymbol{E} > \mathbf{1})$$

Accoding to the inequation above:
If $\boldsymbol{C} \cdot \boldsymbol{D}$, $\boldsymbol{Result} = \mathbf{1}$ iff $\mathbf{1.1} * \boldsymbol{A} + \mathbf{3.1} * \boldsymbol{B} + \mathbf{0.5} * \boldsymbol{E} > \mathbf{4}$ iff $\boldsymbol{A} \cdot \boldsymbol{B}$
If $\boldsymbol{C} \cdot \overline{\boldsymbol{D}}$, $\boldsymbol{Result} = \mathbf{1}$ iff $\mathbf{1.1} * \boldsymbol{A} + \mathbf{3.1} * \boldsymbol{B} + \mathbf{0.5} * \boldsymbol{E} > \mathbf{2}$ iff $\boldsymbol{B}$
If $\overline{\boldsymbol{C}} \cdot \boldsymbol{D}$, $\boldsymbol{Result} = \mathbf{1}$ iff $\mathbf{1.1} * \boldsymbol{A} + \mathbf{3.1} * \boldsymbol{B} + \mathbf{0.5} * \boldsymbol{E} > \mathbf{3}$ iff $\boldsymbol{B}$
If $\overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}}$, $\boldsymbol{Result} = \mathbf{1}$ iff $\mathbf{1.1} * \boldsymbol{A} + \mathbf{3.1} * \boldsymbol{B} + \mathbf{0.5} * \boldsymbol{E} > \mathbf{1}$ iff $\boldsymbol{A} + \boldsymbol{B}$

So:

$$\begin{aligned}
\boldsymbol{Result} &= \boldsymbol{C} \cdot \boldsymbol{D} \cdot \boldsymbol{A} \cdot \boldsymbol{B} + \boldsymbol{C} \cdot \overline{\boldsymbol{D}} \cdot \boldsymbol{B} + \overline{\boldsymbol{C}} \cdot \boldsymbol{D} \cdot \boldsymbol{B} + \overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} \cdot (\boldsymbol{A} + \boldsymbol{B}) \\
&= \boldsymbol{A} \cdot \boldsymbol{B} \cdot \boldsymbol{C} \cdot \boldsymbol{D} + \boldsymbol{A} \cdot \overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{B} \cdot (\overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{C} \cdot \overline{\boldsymbol{D}} + \overline{\boldsymbol{C}} \cdot \boldsymbol{D}) \\
&= \boldsymbol{A} \cdot \boldsymbol{B} \cdot \boldsymbol{C} \cdot \boldsymbol{D} + (\boldsymbol{A} + \mathbf{1}) \cdot \boldsymbol{B} \cdot (\overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{C} \cdot \overline{\boldsymbol{D}} + \overline{\boldsymbol{C}} \cdot \boldsymbol{D}) + \boldsymbol{A} \cdot \overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} \\
&= \boldsymbol{A} \cdot \boldsymbol{B} \cdot \boldsymbol{C} \cdot \boldsymbol{D} + \boldsymbol{A} \cdot \boldsymbol{B} \cdot (\overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{C} \cdot \overline{\boldsymbol{D}} + \overline{\boldsymbol{C}} \cdot \boldsymbol{D}) + \boldsymbol{B} \cdot (\overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{C} \cdot \overline{\boldsymbol{D}} + \overline{\boldsymbol{C}} \cdot \boldsymbol{D}) + \boldsymbol{A} \cdot \overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} \\
&= \boldsymbol{A} \cdot \boldsymbol{B} \cdot (\boldsymbol{C} \cdot \boldsymbol{D} + \overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{C} \cdot \overline{\boldsymbol{D}} + \overline{\boldsymbol{C}} \cdot \boldsymbol{D}) + \boldsymbol{A} \cdot \overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{B} \cdot (\overline{\boldsymbol{C}} + \overline{\boldsymbol{D}}) \\
&= \boldsymbol{A} \cdot \boldsymbol{B} + \boldsymbol{A} \cdot \overline{\boldsymbol{C}} \cdot \overline{\boldsymbol{D}} + \boldsymbol{B} \cdot (\overline{\boldsymbol{C}} + \overline{\boldsymbol{D}})
\end{aligned}$$

3 PROBLEM 3

As Figure 3.1 shows, we can use Breadth First Search to find the minimal set of training instances. The depth of the search tree is the size of the training set, so we can always get the minimal set.

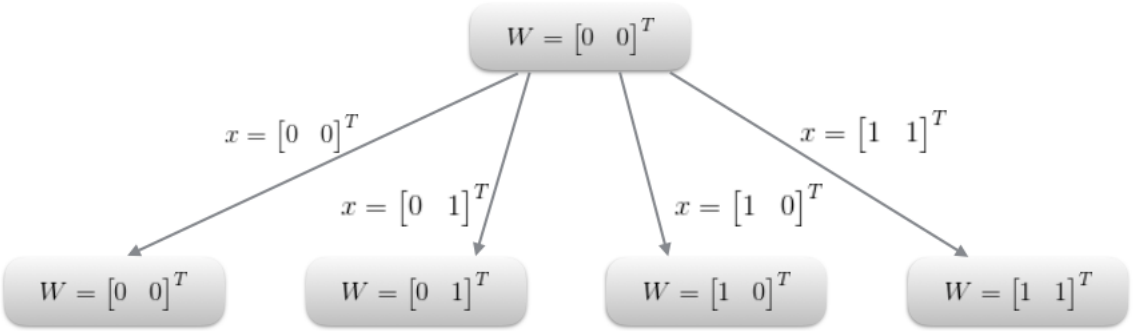

Figure 3.1: Breadth-first search to do error-correction

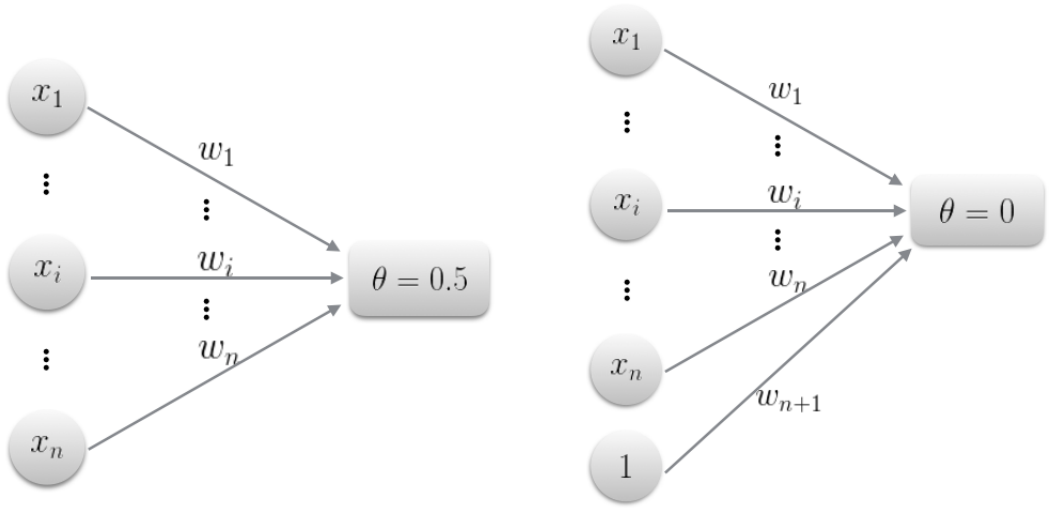

Figure 3.2: TLU without augmented vectors and TLU with augmented vectors

3.1 WITHOUT AUGMENTED VECTORS

As the left part of Figure 3.2 show, we have a preassigned $\boldsymbol{\theta} = \mathbf{0.5}$.

We have learning rate $\boldsymbol{c} = \mathbf{1}$, and $\boldsymbol{w}$ is inited as $\boldsymbol{w_0} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \mathbf{...} & \mathbf{0} & \mathbf{0}\end{bmatrix}$.

If $\boldsymbol{x_1} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0}\end{bmatrix}$, then $\boldsymbol{d_1} = \mathbf{0}$, $\boldsymbol{f_1} = \boldsymbol{w_0 x_1^T} = \mathbf{0} \leqslant \mathbf{0.5}$, $\boldsymbol{w_1} = \boldsymbol{w_0} + \mathbf{1} * (\mathbf{0} - \mathbf{0}) * \boldsymbol{x_1} = \boldsymbol{w_0}$, $\boldsymbol{w}$ didn't change, the model is not improved.

If $\boldsymbol{x_1} = \begin{bmatrix}\boldsymbol{x_{1,1}} & \boldsymbol{x_{1,2}} & \cdots & \boldsymbol{x_{1,n-1}} & \boldsymbol{x_{1,n}}\end{bmatrix}$ $(\exists \boldsymbol{i} \in \boldsymbol{[1,n]}, \boldsymbol{x_{1,i}} \neq \mathbf{0})$, then $\boldsymbol{d_1} = \mathbf{1}$, $\boldsymbol{f_1} = \boldsymbol{w_0 x_1^T} = \mathbf{0} \leqslant \mathbf{0.5}$, $\boldsymbol{w_1} = \boldsymbol{w_0} + \mathbf{1} * (\mathbf{1} - \mathbf{0}) * \boldsymbol{x_1} = \boldsymbol{x_1}$.

We can find that, if $\boldsymbol{x_1} = \begin{bmatrix}\mathbf{1} & \mathbf{1} & \cdots & \mathbf{1} & \mathbf{1}\end{bmatrix} = \boldsymbol{w_1}$, all inputs can get the correct result.

So the minimal set of training instances is $\{\begin{bmatrix}\mathbf{1} & \mathbf{1} & \cdots & \mathbf{1} & \mathbf{1}\end{bmatrix}\}$

3.2 WITH AUGMENTED VECTORS

As the right part of Figure 3.2 show, we have a constant input $\forall \boldsymbol{i}, \boldsymbol{x_{i,n+1}} = \mathbf{1}$.

We have learning rate $\boldsymbol{c} = \mathbf{1}$, and a default threshold $\boldsymbol{\theta} = \mathbf{0}$, and $\boldsymbol{w} \in \boldsymbol{R^{n+1}}$ is inited as $\boldsymbol{w_0} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \mathbf{...} & \mathbf{0} & \mathbf{0}\end{bmatrix}$.

If $\boldsymbol{x_1} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{1}\end{bmatrix}$, then $\boldsymbol{d_1} = \mathbf{0}$, $\boldsymbol{f_1} = \boldsymbol{w_0 x_1^T} = \mathbf{0} \leqslant \mathbf{0}$, $\boldsymbol{w_1} = \boldsymbol{w_0} + \mathbf{1} * (\mathbf{0} - \mathbf{0}) * \boldsymbol{x_1} = \boldsymbol{w_0}$, $\boldsymbol{w}$ didn't change, the model is not improved.

If $x_1 = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} & x_{1,n+1} \end{bmatrix}$ $(\exists i \in [1,n], x_{1,i} \neq 0)$, then $d_1 = 1$, $f_1 = w_0 x_1^T = 0 \leqslant 0$, $w_1 = w_0 + 1 * (1-0) * x_1 = x_1$.

We can find that, if $x_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \end{bmatrix} = w_1$, all inputs except $\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix}$ can get the correct result.

So let $x_2 = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix}$, then $d_2 = 0$, $f_2 = w_1 x_2^T = 1 > 0$, $w_2 = w_1 + 1 * (0-1) * x_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 & 0 \end{bmatrix}$.

Here w_2 perfectly output the "or" result for any input.

So the minimal set of training instances is $\{\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \end{bmatrix}\}$

4 Problem 4

4.1

The input of the fitness function is an algorithm that moves the elevator to take people to the correct floor.
The output of the fitness function is total floors the elevator moved in the process of moving everyone to their target floor.
Algorithms with less total floor should be selected as the next generation.

4.2

The input of the fitness function is the algorithm that changes the traffic lights at the intersection.
The output of the fitness function is total time all vehicles and passerbys waited at the crossroad.
Algorithms with less total time should be selected as the next generation.

5 Problem 5

Feature:
F_1: cricket is on the threshold, $\overline{F_1}$ means the cricket is away from the threshold
OK: ok bit used to remember whether it's ok inside
Action:
A_1: go to the cricket and move it to the threshold
A_2: go inside and check, then return to the threshold
A_3: drag the cricket inside

Production System:

$$\overline{F_1} \longrightarrow A_1, OK = 0$$
$$F_1 \cdot \overline{OK} \longrightarrow A_2, OK = 1$$
$$F_1 \cdot OK \longrightarrow A_3$$