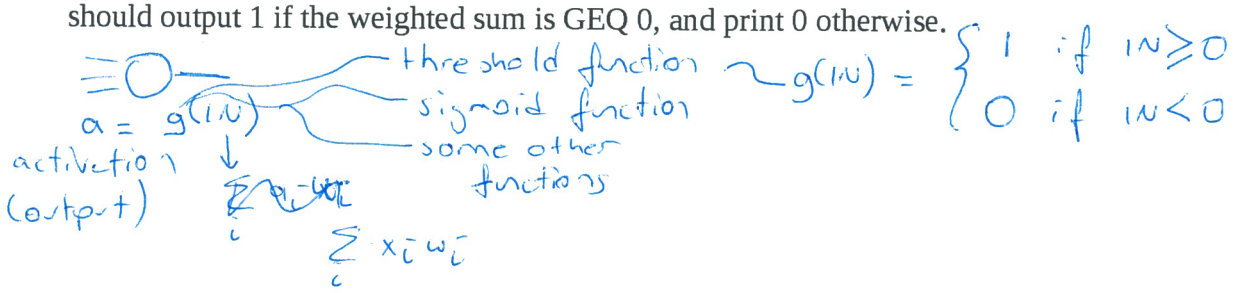
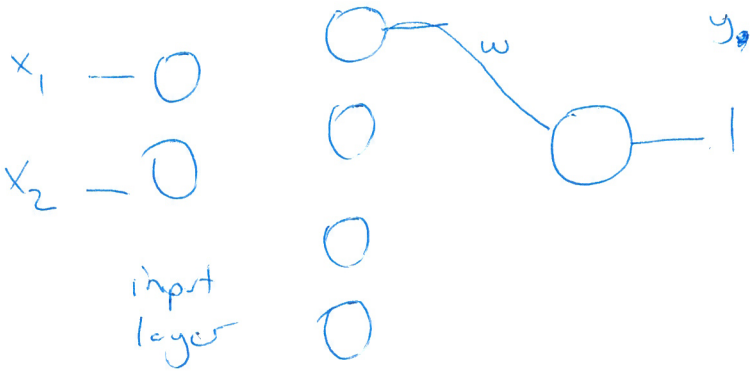
CS4811
January 27, 2016
Wednesday

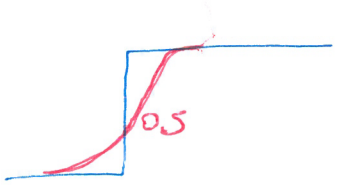
- No class on Monday (February 1st): I will be travelling to a conference.

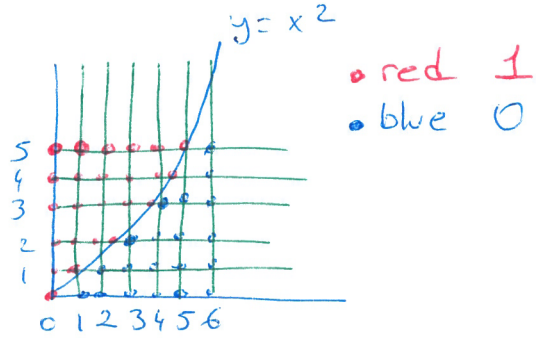
- Remember to use the threshold function to check the neural network. The threshold function should output 1 if the weighted sum is GEQ 0, and print 0 otherwise.



- You need to cast y = x^2 as a classification problem. You should not be trying to create a network that outputs y=x^2 given x.
  One method is to systematically generate point on a square. Then label, the ones one and above the x^2 curve positive (1), and the others negative (0).







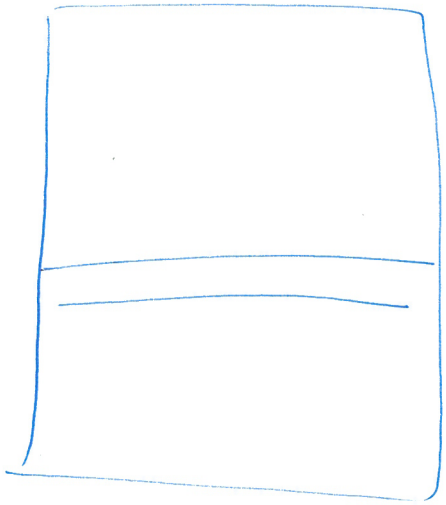
x_1	x_2	y_ϕ
0,	0	1
1,	0	0
2,	0	0
2,	1,	0
⋮		
1,	5	1

CS4811 (2)
Jan 22, 2016
Friday (1)

Search "problem"

1. initial state s
2. actions (ACTIONS(s)) what actions are available
3. transition model (RESULT(s, a)) result of executing action a in state s
4. goal (GOAL-TEST(s)) → state → yes, reached the goal in s / no
5. path cost (optional, additive)

s: state a: action



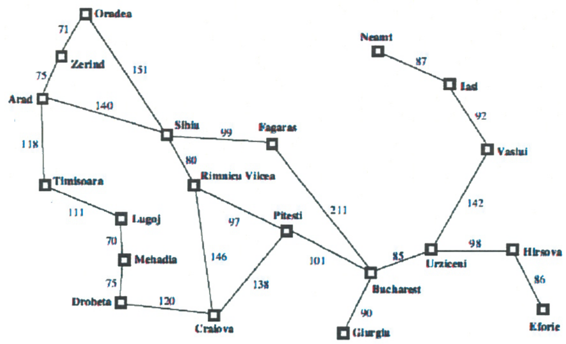
ACTIONS(s) a_1 a_2 a_3
RESULT(s, a)
GOALTEST(s)
yes no

search systematically

We are trying to create a framework that can utilize any search technique and can solve any search problem as long as the ACTIONS, RESULT, GOALTEST functions are implemented.

4

Distances between cities in Romania



Tree search algorithms (cont'd)

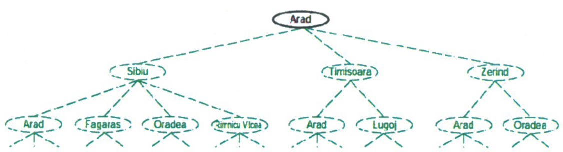
function TREE-SEARCH (*problem, strategy*)
returns a solution, or failure

initialize the frontier using the initial state of *problem*
loop do
  if the frontier is empty **then return** failure
  choose a leaf node and remove it from the frontier
  if the node contains a goal state
    then return the corresponding solution
  expand the chosen node and add the resulting nodes to the frontier
end

arad

frontier
queue of
nodes.

Tree search example

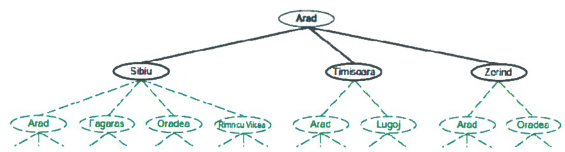


Tim Zerind

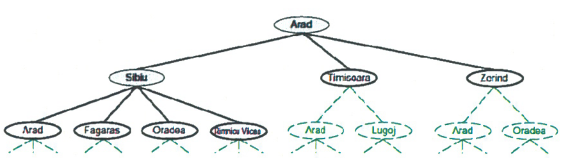
Sibiu Tim Zerind

arad ACTIONS(state = arad)

Tree search example



Tree search example



Graph search algorithms (cont'd)

function GRAPH-SEARCH (*problem*)
returns a solution, or failure

initialize the frontier using the initial state of *problem*
→ initialize the explored set to be empty
loop do
  if the frontier is empty **then return** failure
  choose a leaf node and remove it from the frontier
  if the node contains a goal state
    then return the corresponding solution
→ add the node to the explored set
  expand the chosen node and add the resulting nodes to the frontier
→   only if not in the frontier or explored set
end

Note: A → shows the lines that are added to the tree search algorithm.