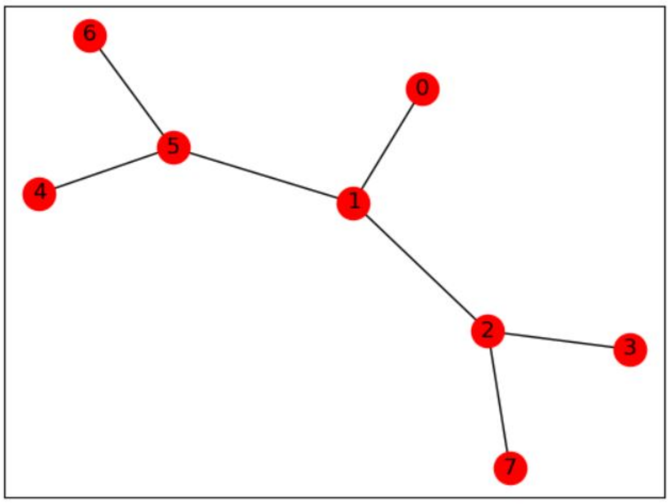
Tutorial – Q learn.



Here we will use Q- Learning to find a path from 0 to 7. WOW !!!

Still, looking through this small code example might give you an intuition about how Q-learning works.

Try it out … And make changes in the code, to improve you understanding of what is going on.

In this simple example, as you can see from the picture shown in the article, the possible choices are all known and the outcome of each choice is deterministic. A best path exists and can be found easily regardless of the initial condition. These nice theoretical hypothesis are usually not true when dealing when real world problems and this makes using Q-learning hard in practice, even though the concept behind it is relatively simple.

Looking at the code in the file, Tutorial_SimpleQLearnGraph.py, we first create a points-list map that represents each direction our bot can take. Using this format allows us to easily create complex graphs but also easily visualize everything with networkx graphs.

The map shows that point **0** is where our bot will start its journey and point **7** is it's final goal. The extra added points and false paths are the obstacles the bot will have to contend with.

We then create the rewards graph - this is the matrix version of our list of points map. We initialize the matrix to be the height and width of our points list (8 in this example) and initialize all values to **-1**:

After that we then build our Q-learning matrix which will hold all the lessons learned from our bot. The Q-learning model uses a transitional rule formula and gamma is the learning parameter.

Try to understand what is going on in the code.

Then try to run it!

Then:

Questions:

a) Change the learning parameter, gamma.. does the algorithm work with all values from 0 to 1 ?
b) In the code we train 700 random points – how few training points are the minimum we can get this to work with?