Question 1

1)
   a) Anytime means that at any given point in the execution of the code, we have the best action to take. The more time we get the better our answer but we always have an answer at any time that is optimal.
   b) In an online situation we are unsure of when the termination condition is. So we have to calculate and be able to act optimally at any point. For a robot for example, if he has to move, we need to know what the best action is. We just don't know when he has to move. Therefore we keep calculating the optimal move until we are “out of time” and have to act.
2) The error is not just the difference between upperbound and lowerbound, but they do contribute to the error. The equation is $E(b^d) = \gamma^d P(b^d) \hat{\varepsilon}(b^d)$ where the upper and lower bounds are defined in the $\hat{\varepsilon}(b^d)$ as $\hat{\varepsilon}(b^d) = U(b) - L(b)$. γ^d is the discount to the power of depth. The only term left is $P(b^d)$. This is the probability of seeing belief. This is given based on how likely the observation is that lead to this belief and how likely the action that led to this is chosen. This leads to the following equation: $P(b^d) = \prod_{i=0}^{d-1} P(o^i \mid b^i, a^i) P(a^i \mid b^i)$. In AEMSII the probability of the action is just set to 1 if it is the best action and 0 otherwise where best action is defined as the action that led to the highest upper bound.
$$P(a \mid b) = \begin{cases} 1 \; if \; a = argmax_{a' \in A} U(a', b) \\ 0 \; otherwise \end{cases}$$
3) To calculate the error term for each leaf belief node, we need to use the equations above. We can see that because of
$$P(a \mid b) = \begin{cases} 1 \; if \; a = argmax_{a' \in A} U(a', b) \\ 0 \; otherwise \end{cases}$$
, the math becomes a lot simpler. The argmax action on belief node b0 is a2. Therefore all the belief nodes under a1 contribute 0 error. For nodes b3 and b4, we will need to do the calculation.

   Error b3 = $\gamma^d P(o^i \mid b^i, a^i) \hat{\varepsilon}(b^d) = .95^1(0.5)(15-9) = 2.85$

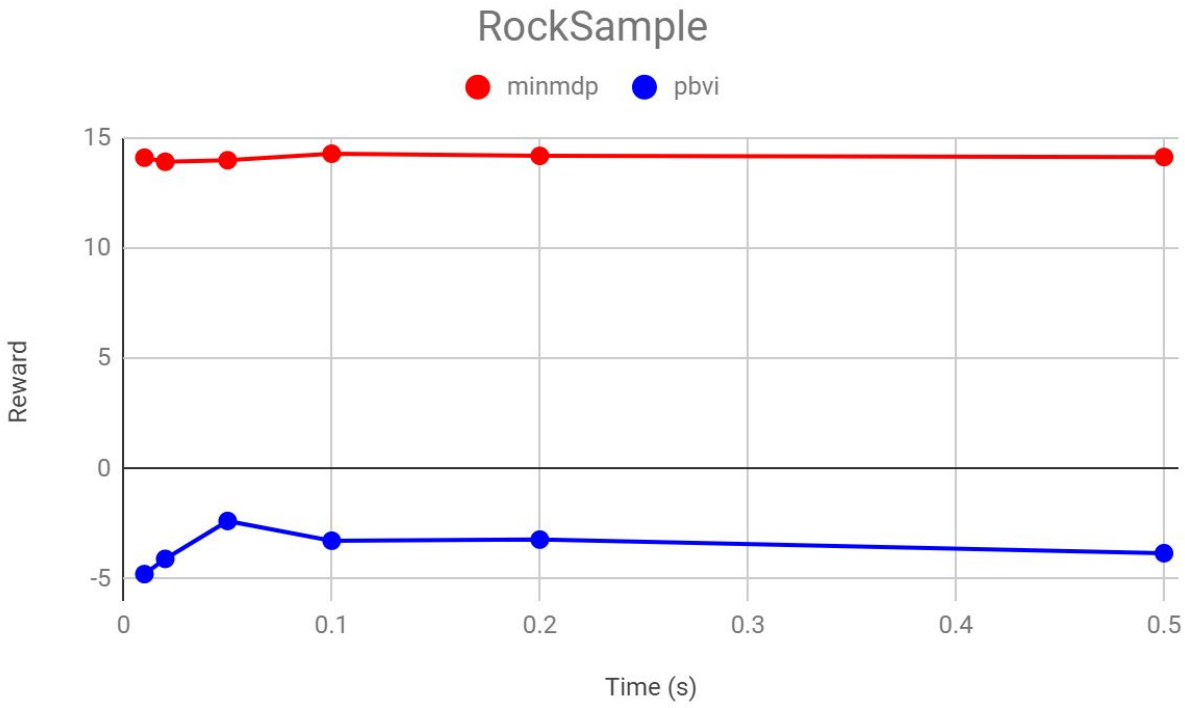
   Error b4 = $\gamma^d P(o^i \mid b^i, a^i) \hat{\varepsilon}(b^d) = .95^1(0.5)(18-10) = 3.8$

   So in total the errors are as follows:
   Error b2 = 0, Error b3 = 2.85, Error b4 = 3.8
   Error b5 = 0, Error b6 = 0, Error b7 = 0, Error b8 = 0

Question 5

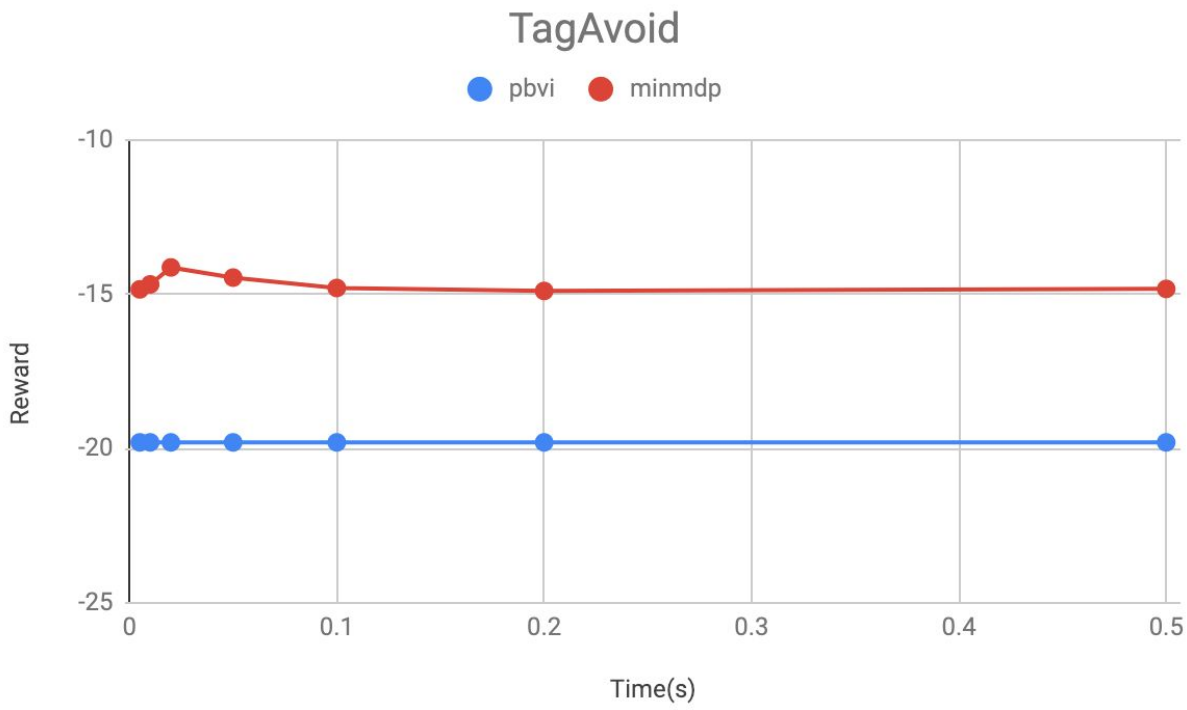
Results

rock sample	pbvi	MinMDP
0.005	-4.785550573	14.09335529
0.01	-4.096143135	13.90951472
0.02	-2.378597658	13.97759712
0.05	-3.272321323	14.27330609
0.1	-3.222281894	14.17603858
0.2	-3.838622773	14.1228792
0.5	-5.800414386	13.80008811



Discussion: I find it very weird that there is no trend in the reward value over time. I would have thought that the more time the algorithm gets the more reward it could find. I am also surprised that we consistently get negative reward for PBVI. I am unsure of the implementation as to how that could cause such an effect.

tagavoid	pbvi	minmdp
0.005	-19.80223271	-14.84715547
0.01	-19.80223271	-14.6857877
0.02	-19.80223271	-14.13663403
0.05	-19.80223271	-14.46573835
0.1	-19.80223271	-14.8026414
0.2	-19.80223271	-14.89813941
0.5	-19.80223271	-14.83203463



Discussion: For the TagAvoid map the running time was a lot longer. At first I thought that this would mean we can find better reward but alas the values are all negative again. As with Rockworld, minMdp performed better. From the first few runs I thought that we would end up with a positive trend for minmdp but that trend died as time increased past the .02 mark.