Case 1:

Two states: initial, finished

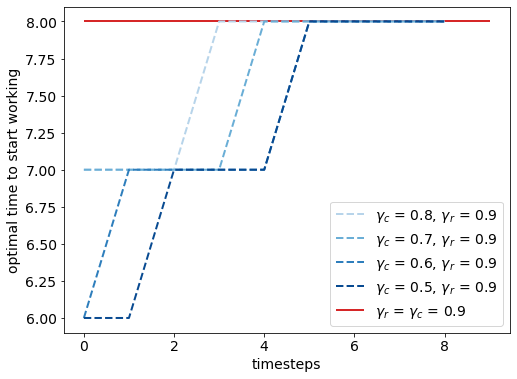
Actions: DO or DON’T in initial state, no choice in finished state

Rewards: effort cost for DO, reward for finished state with a delay at the deadline (no distraction rewards). This is a minimal version of the first case we discussed previously.

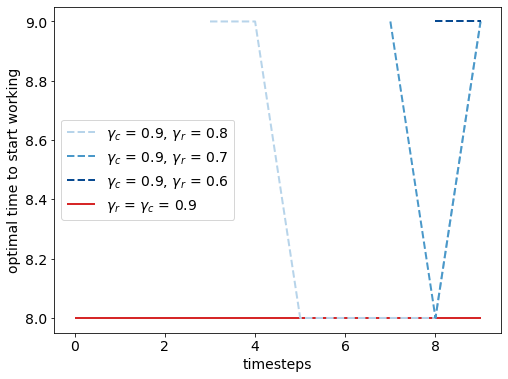
Transitions: Probability of completing on Doing

Parameters: reward\_completed = 4, effort\_do = -1, efficacy = 0.7

When γcost = γreward, there is only planned procrastination. When γcost < γreward, there are defections (to delay more than what was planned) due to preference reversals:



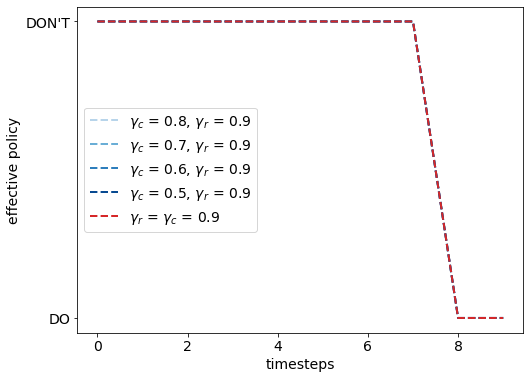
When γcost > γreward, there is the opposite kind of reversals where the agent starts earlier than initially planned or starts despite planning on abandoning.



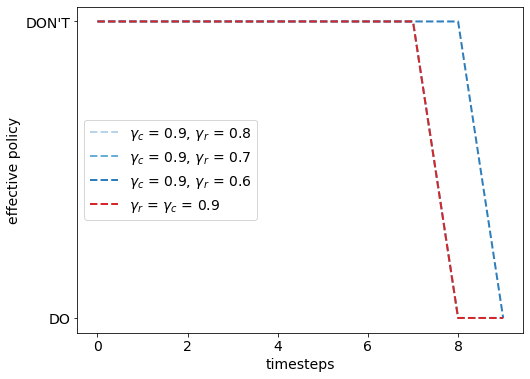
The previous plots show how the policy for the remaining timesteps changes in time.

Here are the plots of the effective policy: the actions to be taken at each timestep (when in state 0) according to current policy

γcost < γreward : identical to policy with equal discounting



γcost > γreward



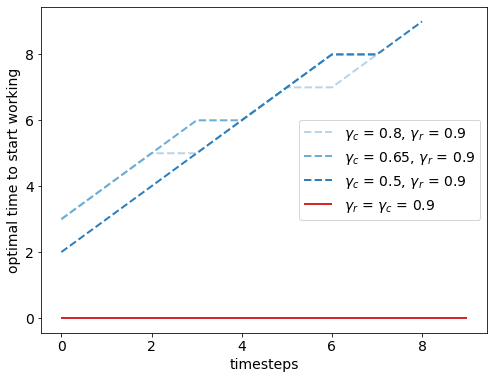
Case 2:

Almost the same as case 1 but immediate rewards on completing instead of delayed rewards.

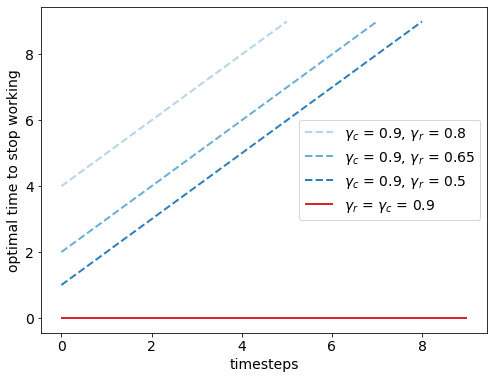
Parameters: reward\_completed = 1.5, effort\_do = -1, efficacy = 0.7

When γcost = γreward, there is no procrastination at all (always optimal to work)

When γcost < γreward, there are planned delays but only with small-ish rewards (with big rewards, it is always worth working despite differential discounting). On top of this, there are defections to delay more than planned before. The reversals are more dynamic here compared to case 1, because reward is also immediate.

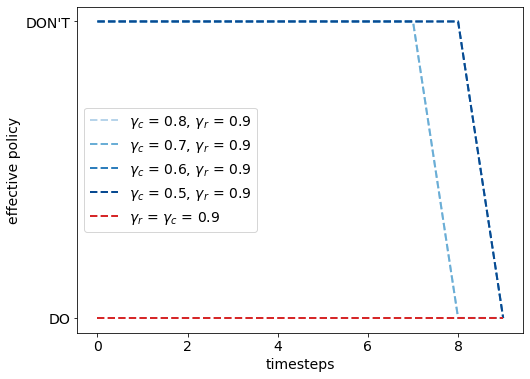


When γcost > γreward, it becomes better to stop working after a point. There are then reversals to stop later than planned.

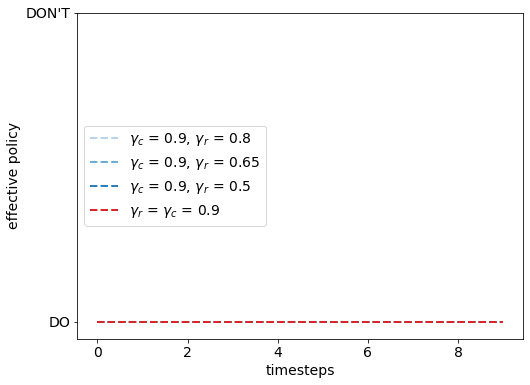


Effective policy

γcost < γreward



γcost > γreward : identical to equal discounting case



Comparison to the LeBouc-Pessiglione policy: Their basic idea was to choose to do the task when the [discounted reward – discounted cost] was the maximum, without forward planning. So only using exponential discounting for rewards and costs doesn’t yield a changing policy. If reward > cost, then it is always optimal to do the task now. They rely on differential hyperbolic discounting of the rewards and costs to get procrastination.

Case 3:

A common hyperbolic discounting scenario where agent must choose to work on one of two tasks -- one with smaller reward but shorter delay and the other larger, far away.

I am still working on this case, it turned out to be a bit more challenging to set up than I had anticipated. I think I might need to have time-varying transitions/ actions (to not allow working on the smaller option after its deadline has passed).