[ ]:
```
import numpy as np
import joblib
from joblib import Parallel, delayed
```

[ ]:
```
#this cell gives the protocol for a single run of fixed-box learning problems

def single_run(y):
    #number of problems per run
    trials = 10000
    #0 of learnurn is WSLS propensity; 1 is reinforcement learning propensity
    learnurn = [1,1]
    #learnrecord records which learning rule was used in each problem
    learnrecord = 0

    for j in range(trials):

        #choice and outcome (success/failure) within each problem
        winrecord = []
        choicerecord = []

        #choice of learning rule determined by learnurn propensities
        #learnchoice == 0 for WSLS; learnchoice == 1 for simple reinforcement
        learnprobs = [learnurn[x]/sum(learnurn) for x in range(2)]
        learnchoice = np.random.choice([0,1], p = learnprobs)
        learnrecord += learnchoice

        #urn is used to determine choice only if learnchoice == 1
        urn = [1,1]

        for i in range(10):

            #if choice determined by WSLS
            if learnchoice == 0:
                if i == 0:
                    choice = np.random.choice([0,1])
                else:
                    if winrecord[-1] == 1:
                        choice = choicerecord[-1]
                    else:
                        choice = 1 - choicerecord[-1]

            #if choice determined by simple reinforcement
            if learnchoice == 1:
                choiceprobs = [urn[x]/sum(urn) for x in range(2)]
                choice = np.random.choice([0,1], p = choiceprobs)

            #record which choice was made and whether that choice was successful
```

```
            #choice == 1, correct box chosen; choice == 0, incorrect box chosen
            choicerecord.append(choice)
            if choice == 1:
                winrecord.append(1)
                #successful act reinforced for reinforcement learning
                urn[choice] += 1
            else:
                winrecord.append(0)

        #reinforce on learning rule used by success rate
        winrate = sum(winrecord)/10
        learnurn[learnchoice] += winrate

    #wslsprop records proportion of problems on which WSLS was used
    wslsprop = 1 - (learnrecord/trials)

    #plus90 records whether the relative frequency of using WSLS >= 0.9 on run
    if wslsprop >= 0.9:
        plus90 = 1
    else:
        plus90 = 0

    #learnurn[0]/sum(learnurn) records final probability of choosing WSLS
    return learnurn[0]/sum(learnurn), plus90
```

[ ]:
```
#this cell runs the simulation

#number of runs
runs = 1000

results = (Parallel(n_jobs = 4)(delayed(single_run)(p) for p in range(runs)))
```

[ ]:
```
#this cell calculates per-run averages of the two single_run statistics

plus90s = []
wsls_probs = []

for p in range(runs):
    wsls_probs.append(results[p][0])
    plus90s.append(results[p][1])

print(plus90s)
print(np.mean(wsls_probs))
print(sum(plus90s)/runs)
```