Simulation Study Vol. 5

In [1]:
```python
from part5_simstudy import *
import warnings

import warnings
warnings.filterwarnings("ignore")
```

Analysis and General Question

5.3.1 Confidence interval width

5_2_1

1. More runs are needed if the alpha is lowered, this makes sense since a lower alpha increses the interval
2. If you increase the sumulation time per run, less runs are needed. In a previous sim study was shown that the influence of the start up phase is smaller when increasing the SIM_Time, the system is run in steady state for longer where the variance is smaller. However, the total simulation time is similar, at least for smaller alpha.

5_2_2

1. The total simulation time neither decreses nor increases by much for a higher BATCH_SIZE.
2. A lower alpha leads to a longer total simulation tieme for the same reason mentioned above.

5_2_1 vs 5_2_2

1. The total simulation time is lower in task_5_2_1 compared to task_5_2_2. The infuluence of the guard pakets is negliglable if set on a resonabe size. If I had ask to be given a prediction, which approach performce better, i.e. needs less time, I would have picked the task_5_2_2. However, the simulation results tell a different story, 5_2_2 simulates longer or just as long as task 5_2_1. In my opinion, the reason for this might be that setting the simulation time to a foxed value and repeating the simulation, leads to very similar results, reagding steady state and the start up phase, i.e. the distributions overall very similar. For the BATCH approach, the frist few batches are taken from the start up phase and then from the staedy state, I think this might be the reason why it needs to simulate longer overall.

In [5]:
```python
task_5_2_1()
```

```
Confidence interval for SIM_TIME: 100000 and alpha: 0.1 is: 0.0014999607944589496
Confidence interval for SIM_TIME: 100000 and alpha: 0.05 is: 0.0014998993912519935
Confidence interval for SIM_TIME: 1000000 and alpha: 0.1 is: 0.0010926124399162384
Confidence interval for SIM_TIME: 1000000 and alpha: 0.05 is: 0.0014989870655033564
SIM TIME:  100s; ALPHA: 10%; NUMBER OF RUNS: 4224; TOTAL SIMULATION TIME (SECONDS): 422400
SIM TIME:  100s; ALPHA:  5%; NUMBER OF RUNS: 6046; TOTAL SIMULATION TIME (SECONDS): 604600
SIM TIME: 1000s; ALPHA: 10%; NUMBER OF RUNS:  2; TOTAL SIMULATION TIME (SECONDS): 2000
SIM TIME: 1000s; ALPHA:  5%; NUMBER OF RUNS:  708; TOTAL SIMULATION TIME (SECONDS): 708000
```

Out[5]:
```
[4224, 6046, 2, 708]
```

In [2]:
```python
task_5_2_2(guard_pkts=10)
```

```
Conf. intervall for BATCHES:100 and alpha: 0.1 is 0.001499950529519799
Conf. intervall for BATCHES:100 and alpha: 0.05 is 0.0014999973897918905
Conf. intervall for BATCHES:1000 and alpha: 0.1 is 0.0014982246066124665
Conf. intervall for BATCHES:1000 and alpha: 0.05 is 0.0014994274680349578
BATCH SIZE:  100; ALPHA: 10%; TOTAL SIMULATION TIME (SECONDS): 447573.735
BATCH SIZE:  100; ALPHA:  5%; TOTAL SIMULATION TIME (SECONDS): 647065.547
BATCH SIZE: 1000; ALPHA: 10%; TOTAL SIMULATION TIME (SECONDS): 475425.385
BATCH SIZE: 1000; ALPHA:  5%; TOTAL SIMULATION TIME (SECONDS): 570864.445
```

Out[2]:
```
[447573735, 647065547, 475425385, 570864445]
```

In [4]:
```python
task_5_2_2(guard_pkts=40)
```

```
Conf. intervall for BATCHES:100 and alpha: 0.1 is 0.0014997623948790994
Conf. intervall for BATCHES:100 and alpha: 0.05 is 0.0014997420768888131
Conf. intervall for BATCHES:1000 and alpha: 0.1 is 0.0014999558814643248
Conf. intervall for BATCHES:1000 and alpha: 0.05 is 0.0014976369038689251
BATCH SIZE:  100; ALPHA: 10%; TOTAL SIMULATION TIME (SECONDS): 444047.017
BATCH SIZE:  100; ALPHA:  5%; TOTAL SIMULATION TIME (SECONDS): 666955.529
BATCH SIZE: 1000; ALPHA: 10%; TOTAL SIMULATION TIME (SECONDS): 477550.015
BATCH SIZE: 1000; ALPHA:  5%; TOTAL SIMULATION TIME (SECONDS): 636593.218
```

Out[4]:
```
[444047017, 666955529, 477550015, 636593218]
```

5.3.2 Confidence interval width

The theoretical blocking probabillity for S=4 and p=0.9 is about 0.126, the average simulation value is between 0.116 to 0.124. The later value is close the the therotical one, it also happens to be the one with 1000s simulation time. The longer the simulation time, the closer the simulation value will e to the theoretical one. AS has been established in a previous simstudy, the value is highly dependend on the influence of the start up phase, it plays less of a role for longer simulations.

In [ ]:
```python
pb = lambda S, p: (1-p)*p**(S+1)/(1-p**(S+2))
S = 4
for p in [0.5, 0.9]:
  print(f"Theoretical blocking prob for S={4} and p={p}: {pb(S, p)}")
```

```
Theoretical blocking prob for S=4 and p=0.5: 0.015873015873015872
Theoretical blocking prob for S=4 and p=0.9: 0.1260225499883686
```

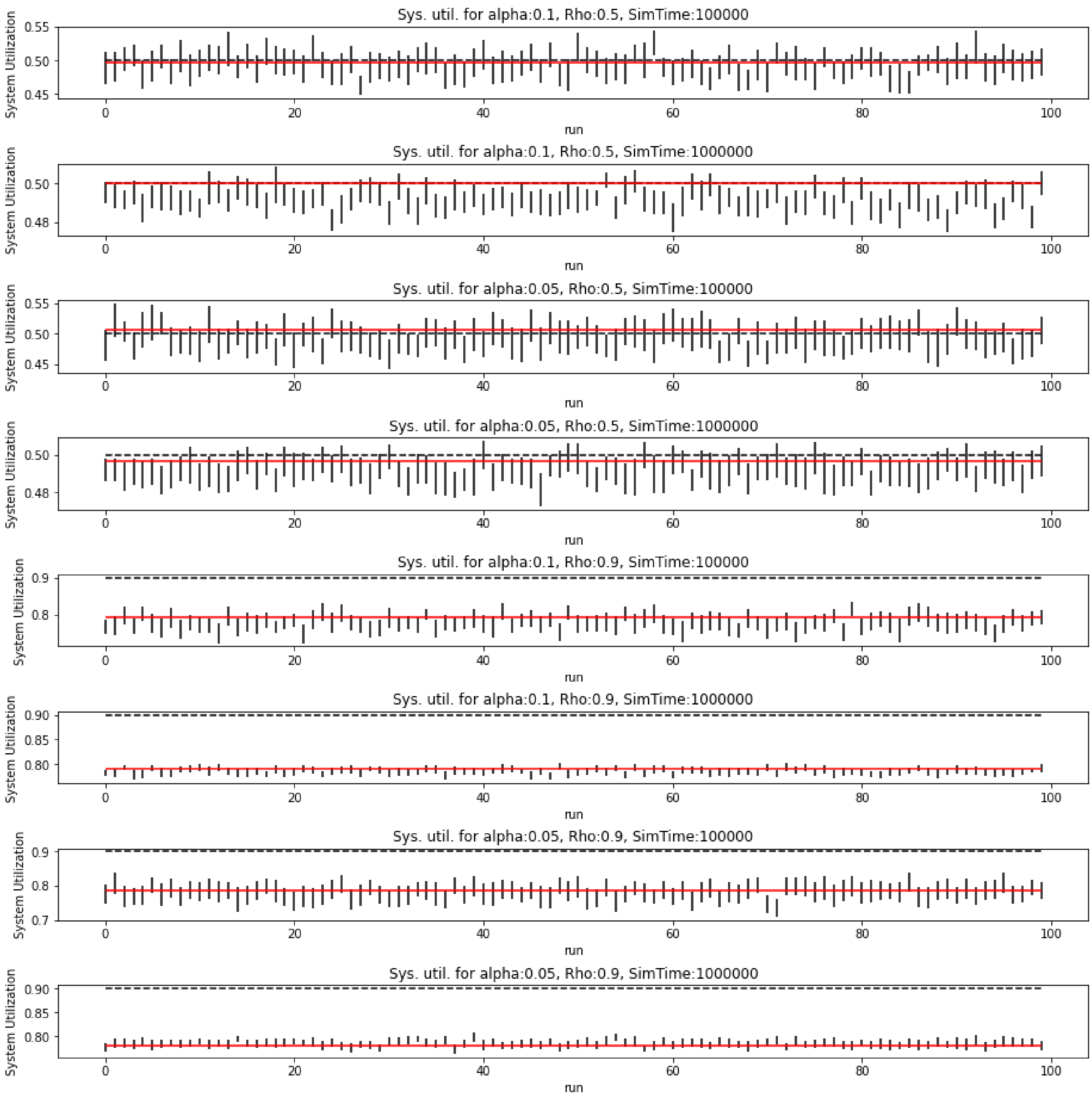
5.3.3 Confidence intervall width

Here are some observations:

1. The system utilization is higher for a higher rho, as it should be.
2. The confidence intervals are smaller for a smaller alpha, also very much expected.
3. For a longer simulation time the actual and theoretical (calc.) mean system utilization devitate less from another. The deviation is higher valued for a rho=0.9
4. The confidence intervals are smaller for a longer simulation time, this is the influence of the warum/start up phase.
5. The actual mean is covered by the intervals most of the times. The calculated mean sometimes is not covered at aöll, this is the influence of the skewness which is introduced by the start up phase.

In [2]:
```python
task_5_2_4()
```

```
act_mean inside interval percentage 0.86
calc_mean inside interval percentage 0.84
act_mean inside interval percentage 0.33
calc_mean inside interval percentage 0.33
act_mean inside interval percentage 0.79
calc_mean inside interval percentage 0.88
act_mean inside interval percentage 0.75
calc_mean inside interval percentage 0.4
act_mean inside interval percentage 0.69
calc_mean inside interval percentage 0.0
act_mean inside interval percentage 0.59
calc_mean inside interval percentage 0.0
act_mean inside interval percentage 0.94
calc_mean inside interval percentage 0.0
act_mean inside interval percentage 0.8
calc_mean inside interval percentage 0.0
```



5.3.4 Variable simulation time

No, if we would only take packets each time a packet is dropeed, we basically from a conditional probability, this does not ensure independence of the test daata, which is an necesary condition for the t test.

Bootstrapping

Bootstrapping does not require additional simulations which can be benefitial in case a siimulation is very costly or time intensive. If we have a single very long simulation and form the confidence interval over it with both approaches, then they are similar.

In [ ]: