Reproducing the experiments

The prototypes of both the proposed sample-driven solving procedure and the isolation-based one in the previous work have been implemented in Python 3.10, running on an Apple M1 chip with 8 cores and 16 GB RAM. Each instance mentioned in Table 4 and Fig. 4 of the manuscript can be reproduced, but there are slight differences in time consumption and space consumption when running on different machines. All data involved in the manuscript are integrated in the file `./RR/QCTMC/paper_results/result.txt`. For reproducing the results, we first move to the folder `RR/QCTMC/` and all subsequent commands should be executed under that folder.

```
cd RR/QCTMC
```

1. computing the exponential of governing matrix

Both the isolation-based algorithm and the sample-driven one require the computation of the general solution `\rho(t)`, and the main time consumption is in computing the exponential of governing matrix, which takes about 15 minutes. Executing the following command will compute the exponential of governing matrix and will store the result in the file `./EXPM.txt`.

```
python3 QCTMC.py 1
```

Then the result stored in file `./EXPM.txt` will be used as an intermediate for calculating `\rho(t)` in both the isolation-based algorithm and the sample-driven one.

2. running the random observing expressions

The data mentioned in Table 4 and Fig. 4 of the manuscript can be batch-calculated by running the following commands, the execution of which takes nearly 3 hours.

```
./shell.sh
```

The seeds of random observing expression used in the manuscript are stored in the folder `./paper_results/pickle_files/`. Each binary file include five instances, and the file is named by the features of the instance such as whether the observing expression is "single signal" or "cnf of multiple signals", the expression's degree and range of height. For example, the observing expressions in the binary file "exp_1 _1 _10.pkl" is "single signal", degree: 1, and the range of height: [1, 10]. The file "bool_exp _1 _1 _10.pkl" is "cnf of multiple signals", degree: 1, and the range of height: [1, 10]. The commands of isolation-based and sample-driven algorithms are integrated in the file `./shell.sh`, which can also be executed in a single command.

For example, a sample-driven algorithm command is as follows.

```
python3 BatchObservingExpression.py 1 1 10 1 0 > results/degree1height10.txt
```

The parameters "1 1 10 1 0" denote degree: 1, the range of height: [1, 10], sample-driven: 1, single signal: 0, and the details of each instance are outputed to file `./results/degree1height10.txt` . The forth parameter denotes the applied algorithm "isolation-based": 0 or "sample-driven": 1, and the fifth parameter denotes "single signal": 0 or "cnf of multiple signals": 1. The summative results are outputed in file `./result.txt` , which has the following format.

```
**SINGLE SIGNAL** SAMPLE-DRIVEN: DEGREE:1, HEIGHT:1-10
ID TIME SPACE SATISFIED
11 3.26 107 0
12 1.09 107 1
13 1.13 107 1
14 1.12 107 1
15 1.09 107 0
AVERAGE TIME: 1.54, AVERAGE SPACE: 107.40
******************END**********************
```

The first row denotes the type of observing expression "single signal" or "cnf of multiple signals", the algorithm selected "isolation-based" or "sample-driven", expression's degree and range of height. The second row is the field names, which includes the ID of instance, time consumption, space consumption, and the boolean value indicating whether the repeated reachability is satisfied. At the end, the seventh row shows the time consumption and space consumption on the average of five instances.

The command of isolation-based algorithm is as follows. The parameters "1 1 10 0 0" denote degree: 1, the range of height: [1, 10], isolation-based: 0, single signal: 0, and the details are added to the end of file `./results/degree1height10.txt` .

```
python3 BatchObservingExpression.py 1 1 10 0 0 >> results/degree1height10.txt
```

The summative results are outputed in file `./result.txt` , which has the follow format.

```
**SINGLE SIGNAL** ISOLATION: DEGREE:1, HEIGHT:1-10
ID TIME SPACE SATISFIED
11 9.57 111 0
12 1.24 112 1
13 2.58 113 1
14 1.42 113 1
15 1.23 113 0
AVERAGE TIME: 3.21, AVERAGE SPACE: 112
******************END**********************
```

In the following table, we list the commands in the file `./shell.sh` , classified by degree, height, algorithm, and the type of observing expression. The symbol "XXX" denotes the parameter "degree".

DEGREE	HEIGHT	ALGORITHM	SINGLE SIGNAL	CNF OF MULTIPLE SIGNALS
XXX	[1,10]	SAMPLE-DRIVEN	python3 BatchObservingExpression.py XXX 1 10 1 0 > results/degreeXXXheight10.txt	python3 BatchObservingExpression.py XXX 1 10 1 1 > results/booldegreeXXXheight10.txt
		ISOLATION-BASED	python3 BatchObservingExpression.py XXX 1 10 0 0 >> results/degreeXXXheight10.txt	python3 BatchObservingExpression.py XXX 1 10 0 1 >> results/booldegreeXXXheight10.txt
	[11,100]	SAMPLE-DRIVEN	python3 BatchObservingExpression.py XXX 11 100 1 0 > results/degreeXXXheight100.txt	python3 BatchObservingExpression.py XXX 11 100 1 1 > results/booldegreeXXXheight100.txt
		ISOLATION-BASED	python3 BatchObservingExpression.py XXX 11 100 0 0 >> results/degreeXXXheight100.txt	python3 BatchObservingExpression.py XXX 11 100 0 1 >> results/booldegreeXXXheight100.txt
	[101,500]	SAMPLE-DRIVEN	python3 BatchObservingExpression.py XXX 101 500 1 0 > results/degreeXXXheight500.txt	python3 BatchObservingExpression.py XXX 101 500 1 1 > results/booldegreeXXXheight500.txt
		ISOLATION-BASED	python3 BatchObservingExpression.py XXX 101 500 0 0 >> results/degreeXXXheight500.txt	python3 BatchObservingExpression.py XXX 101 500 0 1 >> results/booldegreeXXXheight500.txt
	[501,1000]	SAMPLE-DRIVEN	python3 BatchObservingExpression.py XXX 501 1000 1 0 > results/degreeXXXheight1000.txt	python3 BatchObservingExpression.py XXX 501 1000 1 1 > results/booldegreeXXXheight1000.txt
		ISOLATION-BASED	python3 BatchObservingExpression.py XXX 501 1000 0 0 >> results/degreeXXXheight1000.txt	python3 BatchObservingExpression.py XXX 501 1000 0 1 >> results/booldegreeXXXheight1000.txt

3. producing random observing expressions

The following command generates five **"single signal"** observing expressions and outputs the summative results of the algorithms "isolation-based" and "sample-driven" in the file `./result.txt` . The details of each instance are in the file `./results/degree3height1000.txt` . The parameters "3 501 1000 0" denote degree: 3, the range of height: [501, 1000], and single signal: 0.

```
python3 ObservingExpression.py 3 501 1000 0 > results/degree3height1000.txt
```

The following command generates five **"cnf of multiple signals"** observing expressions and outputs the summative results of the algorithms "isolation-based" and "sample-driven" in the file `./result.txt` . The details of each instance are in the file `./results/booldegree3height1000.txt` . The parameters "3 501 1000 1" denote degree: 3, the range of height: [501, 1000], cnf of multiple signals: 1.

```
python3 ObservingExpression.py 3 501 1000 1 > results/booldegree3height1000.txt
```