A Technique for Object-Centric Compliance Checking

Technical Report

1 Performance Analysis

In this section, we describe the steps required to reproduce the performance analysis of the application of Object-Centric Compliance Rules (OCCRs) with the tool provided as a by-product of our work. The tool source code, the event logs used for the evaluation, and the scripts for the presentation of the results are available at `https://anonymous.4open.science/r/8601`.

1.1 Environment Setup

To ensure the replicability of the performance analysis, the application can be installed using Docker or through a manual setup. The Docker installation provides a streamlined approach by encapsulating dependencies within a container, while the manual setup offers more flexibility for modifications.

Installation with Docker. The installation with Docker requires having Docker installed on the system. The following commands allow building the necessary image and running the application in a containerized environment:

```
$ docker build -t occc .
$ run -p 8000:8000 occc
```

Once the container is running, the local instance of the application can be accessed at `http://0.0.0.0:8000/`.

Manual Installation. For a manual installation, the setup requires both Node.js and Python. The installation proceeds in two stages: frontend and backend.

For the frontend, the required packages must be installed, and the development server must start using:

```
$ ./frontend npm i
$ ./frontend npm run dev
```

For the backend, Python dependencies need to be installed before starting the service:

```
$ ./backend pip install -r requirements.txt
$ ./backend uvicorn app:app -reload
```

Once both the frontend and backend are running, the application can be accessed locally at `http://localhost:5173/`.

1.2 Tests Execution

The performance analysis is conducted using `pytest` and `pytest-benchmark`. The test suite is structured into separate directories, each targeting a different synthetic log under analysis:

```
tests/
|-- data/
|    |-- 2000/
|    |-- 10000/
|-- test_logistic/
|    |-- test_logistic.py
|-- test_om/
|    |-- test_om.py
|-- test_p2p/
|    |-- test_p2p.py
|-- test_logistic_ext/
|    |-- test_logistic_ext.py
|-- results/
```

More precisely, the *data* folder contains two sub-folders, the *2000* folder which contains the three events log divided into chunks of 2000 events with the corresponding events, and the *10000* folder which contains the *Logistics* log extended to 100,000 total events and divided in chunks of 10,000 events per log. The *test_ ** folders contain the specific test suite for the corresponding event log.

Running the Tests. Each of the *test_ *.py* files contains specific test cases, i.e., a set of OCCRs in the log's domain, to analyze the scalability of the OCCRs application's performance. The defined OCCRs follow the same pattern for each event log. They are defined with an increasing complexity which causes higher computation time. Specifically, the rules have the following characteristics:

R1 are unary OCCRs, thus involving a unique event pattern and a unary control-flow pattern,

R2 are binary OCCRs, where the two event patterns present constraint that limits the cardinality of the set of events resulting from the application of the event pattern $eval_P$,

R3 are the same binary OCCRs of R2 without the constraint in the event pattern, thus considering a wider number of events resulting from the application of the event pattern $eval_P$,

R4 and R5 reflect the same structure of R2 and R3, but in this case, the OCCRs employ a more complex event linkage ψ, i.e., with a cardinality strictly greater than 1; while R2 and R3 have a simple event linkage of cardinality equal to 1,

R6 is a binary OCCR that employs the *xLeadsTo* binary control-flow pattern. We chose to include this specific control-flow pattern because, based on its application definition, it is the most computationally expensive.

The tests must be executed individually for each type of event log. The following commands launch the performance tests and store the benchmark results in JSON files:

```
$ pytest tests/test_logistic -benchmark-json tests/results/
benchmark_results_log.json
$ pytest tests/test_om -benchmark-json tests/results/
benchmark_results_om.json
$ pytest tests/test_p2p -benchmark-json tests/results/
benchmark_results_p2p.json
$ pytest tests/test_logistic_ext -benchmark-json tests/results/
benchmark_results_log_ext.json
```

These results will be used for further analysis in the subsequent phases of the performance analysis.

1.3 Results Processing

After executing the performance tests, `pytest-benchmark` produces detailed benchmark tables summarizing key performance metrics for each test case. These tables reported in Figures 1 to 3, provide insights into the execution time distribution across multiple iterations. Notably, the three event logs present a different number of tests due to their size which consequence is a different number of event log chunks.

Each table includes several performance indicators: the *minimum* and *maximum* execution times observed, the *mean* execution time computed over all iterations, and the *standard deviation*, which quantifies variability in execution time. Additionally, the *median* value is reported, offering a robust measure of central tendency, along with the *interquartile range* (IQR), which captures the dispersion of execution times. The number of *outliers* is also identified, indicating runs that significantly deviated from the typical performance. Another key metric is the *operations per second* (OPS), which reflects the number of executions performed within a given time frame, providing an inverse measure of execution time efficiency. Finally, the table reports the total number of *rounds* (i.e., repetitions of the benchmark) and the number of *iterations* within each round, ensuring statistical robustness in the performance assessment.

To facilitate further analysis, these raw benchmark results are stored in JSON files, generated separately for each test case. These files contain structured data that allow for manual processing and visualization. Two Python notebooks are used to extract relevant information, reformat the results, and generate additional performance visualizations tailored to the specific needs of this evaluation. The notebooks are available at `https://anonymous.4open.science/r/8601/backend/tests/processing/` and must be run after the test executions.

1.4 Results Presentation

After executing the tests and processing the benchmark results, the extracted performance data is structured into a summary tables, see Tables 1 and 2. These

benchmark: 102 tests

Name (time in us)	Min	Max	Mean	StdDev	Median	IQR	Outliers	OPS	Rounds	Iterations
test_R1_log[dataset0]	62.9160 (1.0)	106.2080 (1.0)	66.2499 (1.0)	3.5331 (1.0)	65.5420 (1.0)	1.0625 (1.0)	152;368	15,094.3530 (1.0)	2164	1
test_R1_log[dataset1]	120.5830 (1.92)	212.5000 (2.00)	125.7707 (1.90)	6.1732 (1.75)	123.6455 (1.89)	4.0000 (3.76)	240;231	7,950.9796 (0.53)	2736	1
test_R1_log[dataset2]	180.0000 (2.86)	242.7090 (2.29)	186.0541 (2.81)	5.1122 (1.45)	185.9160 (2.84)	5.2920 (4.98)	249;68	5,374.7810 (0.36)	2051	1
test_R1_log[dataset3]	236.0830 (3.75)	342.5830 (3.23)	245.3964 (3.70)	7.7453 (2.19)	244.8335 (3.74)	6.5420 (6.16)	269;116	4,075.0389 (0.27)	1898	1
test_R1_log[dataset4]	292.8750 (4.66)	473.5830 (4.46)	303.9123 (4.59)	8.2058 (2.32)	304.0410 (4.64)	6.1348 (5.77)	189;60	3,290.4225 (0.22)	1495	1
test_R1_log[dataset5]	351.5420 (5.59)	594.0000 (5.59)	363.8971 (5.49)	10.9748 (3.11)	363.3750 (5.54)	7.5830 (7.14)	102;54	2,748.0295 (0.18)	1266	1
test_R1_log[dataset6]	410.8750 (6.53)	702.4160 (6.61)	426.3688 (6.44)	15.1300 (4.28)	426.0415 (6.50)	9.7920 (9.22)	43;34	2,345.3874 (0.16)	1040	1
test_R1_log[dataset7]	469.6250 (7.46)	59,119.3340 (556.64)	642.7049 (9.70)	2,180.9078 (617.28)	489.5415 (7.47)	13.6240 (12.82)	8;115	1,555.9241 (0.10)	918	1
test_R1_log[dataset8]	531.1250 (8.44)	1,507.7500 (14.20)	572.5531 (8.64)	65.3028 (18.48)	556.3955 (8.49)	21.2500 (20.00)	48;91	1,746.5628 (0.12)	822	1
test_R1_log[dataset9]	591.2500 (9.40)	1,049.3750 (9.88)	623.9089 (9.42)	29.1170 (8.24)	618.3335 (9.43)	8.3340 (7.84)	68;154	1,602.7981 (0.11)	726	1
test_R1_log[dataset10]	661.4580 (10.51)	1,180.3750 (11.11)	695.9439 (10.50)	37.3516 (10.57)	689.9170 (10.53)	15.0000 (14.12)	50;76	1,436.8974 (0.10)	690	1
test_R1_log[dataset11]	740.2500 (11.77)	16,617.1660 (156.46)	1,146.5653 (17.31)	1,450.3490 (410.50)	833.5840 (12.72)	207.8023 (195.58)	6;18	872.1701 (0.06)	157	1
test_R1_log[dataset12]	787.1670 (12.51)	1,837.5000 (17.30)	861.5655 (13.00)	113.2468 (32.05)	825.9580 (12.60)	37.9378 (35.71)	29;44	1,160.6778 (0.08)	325	1
test_R1_log[dataset13]	863.1670 (13.72)	1,501.0840 (14.13)	929.9532 (14.04)	73.8065 (20.89)	904.1670 (13.80)	24.2610 (22.83)	48;96	1,075.3229 (0.07)	535	1
test_R3_log[dataset0]	881.7500 (14.01)	1,055.5410 (9.94)	919.8102 (13.88)	25.1602 (7.12)	913.1660 (13.93)	16.0310 (15.09)	51;29	1,087.1809 (0.07)	259	1
test_R2_log[dataset0]	884.7080 (14.06)	3,758.0830 (35.38)	1,075.7498 (16.24)	341.5555 (96.67)	958.7920 (14.63)	132.1875 (124.41)	83;121	929.5842 (0.06)	880	1
test_R1_log[dataset14]	945.3330 (15.03)	1,529.0000 (14.40)	1,010.9622 (15.26)	68.7753 (19.47)	984.2920 (15.02)	41.5102 (39.07)	54;54	989.1567 (0.07)	453	1
test_R1_log[dataset15]	1,020.9160 (16.23)	2,531.8330 (23.84)	1,108.6380 (16.73)	115.2958 (32.63)	1,070.3125 (16.33)	73.3960 (69.08)	33;23	902.0077 (0.06)	408	1
test_R1_log[dataset16]	1,105.4170 (17.57)	1,905.8750 (17.94)	1,165.0986 (17.59)	65.2515 (18.47)	1,151.5420 (17.57)	19.5825 (18.43)	27;51	858.2964 (0.06)	359	1
test_R3_log[dataset1]	1,838.3340 (29.22)	4,814.3750 (45.33)	1,930.0224 (29.13)	181.8665 (51.48)	1,906.0625 (29.08)	33.6455 (31.67)	8;41	518.1287 (0.03)	452	1
test_R2_log[dataset1]	1,893.2910 (30.09)	3,962.5420 (37.31)	1,963.5324 (29.64)	161.8916 (45.82)	1,921.6045 (29.32)	50.6250 (47.65)	20;44	509.2862 (0.03)	442	1
test_R3_log[dataset2]	3,139.5420 (49.90)	4,727.5830 (44.51)	3,265.8780 (49.30)	122.5418 (34.68)	3,242.0420 (49.47)	45.8750 (43.18)	13;28	306.1964 (0.02)	276	1
test_R2_log[dataset2]	3,166.9170 (50.34)	3,694.1670 (34.78)	3,285.8354 (49.60)	74.9713 (21.22)	3,262.1670 (49.77)	54.7502 (51.53)	33;25	304.3366 (0.02)	259	1
test_R3_log[dataset3]	4,498.6250 (71.50)	4,809.7500 (45.29)	4,653.7868 (70.25)	41.8040 (11.83)	4,659.2910 (71.09)	31.1562 (29.32)	35;20	214.8788 (0.01)	197	1
test_R2_log[dataset3]	4,593.9160 (73.02)	38,871.3330 (365.99)	5,211.7994 (78.67)	3,017.2161 (853.99)	4,667.2920 (71.21)	60.8537 (57.27)	7;36	191.8723 (0.01)	193	1
test_R3_log[dataset4]	6,021.9590 (95.71)	6,286.3330 (59.19)	6,200.8291 (93.60)	39.3312 (11.13)	6,202.9580 (94.64)	31.9162 (30.04)	23;13	161.2688 (0.01)	147	1
test_R2_log[dataset4]	6,165.1670 (97.99)	7,069.4580 (66.56)	6,379.5375 (96.29)	205.8439 (58.26)	6,311.7080 (96.30)	267.6980 (251.95)	26;5	156.7512 (0.01)	141	1
test_R3_log[dataset5]	8,085.8330 (128.52)	8,503.2500 (80.06)	8,283.8976 (125.04)	61.9181 (17.53)	8,274.5000 (126.25)	55.9487 (52.66)	24;8	120.7161 (0.01)	111	1
test_R2_log[dataset5]	8,283.9580 (131.67)	11,135.2080 (104.84)	8,653.0092 (130.61)	553.3026 (156.61)	8,397.1670 (128.12)	351.4067 (330.74)	15;15	115.5667 (0.01)	109	1
test_R3_log[dataset6]	10,210.9170 (162.29)	10,871.5830 (102.36)	10,465.0791 (157.96)	79.1744 (22.41)	10,467.6250 (159.71)	55.3445 (52.09)	10;6	95.5559 (0.01)	89	1
test_R2_log[dataset6]	10,447.5000 (166.05)	16,298.3330 (153.46)	11,261.8763 (169.99)	1,167.1208 (330.34)	10,869.1670 (165.84)	758.6980 (714.07)	6;6	88.7952 (0.01)	65	1
test_R5_log[dataset0]	12,579.4590 (199.94)	16,525.0000 (155.59)	12,884.1240 (194.48)	654.2430 (185.18)	12,673.6250 (193.37)	182.9063 (172.15)	4;10	77.6149 (0.01)	77	1
test_R4_log[dataset0]	12,582.6670 (199.99)	14,708.6250 (138.49)	12,932.8427 (195.21)	480.8060 (136.09)	12,687.5835 (193.58)	524.8965 (494.02)	10;4	77.3225 (0.01)	72	1
test_R3_log[dataset7]	12,715.5830 (202.10)	13,498.2910 (127.09)	12,871.9760 (194.29)	123.3955 (34.93)	12,857.6250 (196.17)	90.0725 (84.77)	8;5	77.6882 (0.01)	73	1
test_R6_log[dataset0]	12,722.0000 (202.21)	13,688.1250 (128.88)	12,877.6537 (194.38)	179.9345 (50.93)	12,814.3330 (195.51)	115.3648 (108.58)	10;10	77.6539 (0.01)	77	1
test_R2_log[dataset7]	12,843.5830 (204.14)	96,529.2500 (908.87)	17,435.0640 (263.17)	12,765.6510 (>1000.0)	14,007.4580 (213.72)	2,501.4370 (>1000.0)	2;7	57.3557 (0.00)	56	1
test_R2_log[dataset8]	15,103.1250 (240.05)	57,521.7500 (541.60)	17,256.9703 (260.48)	8,160.6842 (>1000.0)	15,344.0835 (234.11)	562.0415 (528.98)	2;7	57.9476 (0.00)	52	1
test_R3_log[dataset8]	15,184.8340 (241.35)	17,348.3750 (163.34)	15,586.2712 (235.26)	447.3662 (126.62)	15,462.2910 (235.91)	108.8012 (102.40)	6;13	64.1590 (0.00)	61	1
test_R2_log[dataset9]	16,185.7500 (257.26)	35,448.5830 (333.77)	16,980.6939 (256.31)	2,662.9394 (753.71)	16,310.4160 (248.85)	503.7397 (474.11)	3;5	58.8904 (0.00)	55	1
test_R2_log[dataset10]	17,536.0420 (278.72)	20,037.7910 (188.67)	18,180.6467 (274.43)	657.1739 (186.01)	17,899.0625 (273.09)	839.1035 (789.75)	9;3	55.0035 (0.00)	52	1
test_R3_log[dataset9]	17,809.5000 (283.07)	18,710.1670 (176.17)	18,106.5440 (273.31)	139.6195 (39.52)	18,100.9375 (276.17)	68.0415 (64.04)	7;8	55.2287 (0.00)	52	1
test_R2_log[dataset11]	18,645.5840 (296.36)	52,478.7500 (494.11)	25,009.1774 (377.50)	8,649.3851 (>1000.0)	20,878.3330 (318.55)	10,549.7395 (>1000.0)	6;2	39.9853 (0.00)	39	1
test_R2_log[dataset12]	19,794.5420 (314.62)	24,437.2500 (230.09)	21,328.1859 (321.94)	1,191.6844 (337.29)	21,077.2080 (321.58)	1,478.0418 (>1000.0)	10;2	46.8863 (0.00)	39	1
test_R2_log[dataset13]	20,901.0840 (332.21)	23,246.3340 (218.88)	21,334.5427 (322.03)	547.8008 (155.05)	21,090.5625 (321.79)	382.7300 (360.22)	7;5	46.8723 (0.00)	44	1
test_R3_log[dataset10]	21,416.6250 (340.40)	45,723.3330 (430.51)	23,018.6060 (347.45)	3,942.9403 (>1000.0)	21,755.1455 (331.93)	875.0000 (823.53)	4;6	43.4431 (0.00)	44	1
test_R2_log[dataset14]	22,406.9160 (356.14)	28,374.1660 (267.16)	24,544.5853 (370.48)	1,328.4116 (375.99)	24,359.0000 (371.65)	1,394.9372 (>1000.0)	11;2	40.7422 (0.00)	41	1
test_R2_log[dataset15]	23,540.5000 (374.16)	25,377.0000 (238.94)	24,106.4999 (363.87)	495.6476 (140.29)	23,872.2500 (364.23)	566.3022 (532.99)	9;2	41.4826 (0.00)	37	1
test_R2_log[dataset16]	24,730.5420 (393.07)	37,374.8330 (351.90)	26,004.8726 (392.53)	2,285.5254 (646.89)	25,137.9580 (383.54)	1,122.4163 (>1000.0)	3;3	38.4543 (0.00)	35	1
test_R3_log[dataset11]	24,906.7080 (395.87)	27,242.7910 (256.50)	25,214.6339 (380.60)	446.6695 (126.42)	25,086.0415 (382.75)	165.0000 (155.30)	4;6	39.6595 (0.00)	38	1
test_R3_log[dataset12]	28,219.1250 (448.52)	29,088.4160 (273.88)	28,504.7235 (430.26)	206.6214 (58.48)	28,464.5840 (434.30)	171.6873 (161.59)	6;4	35.0819 (0.00)	33	1
test_R3_log[dataset13]	31,884.9170 (506.79)	33,556.2920 (315.95)	32,205.1374 (486.12)	403.6224 (114.24)	32,057.8540 (489.12)	315.7080 (297.14)	2;2	31.0509 (0.00)	30	1
test_R3_log[dataset14]	36,593.0410 (581.62)	41,182.3750 (387.75)	37,700.6507 (569.07)	1,180.3981 (334.10)	37,248.9790 (568.32)	1,182.2090 (>1000.0)	4;2	26.5247 (0.00)	26	1
test_R4_log[dataset1]	41,291.7080 (656.30)	42,718.5830 (402.22)	41,788.7794 (630.77)	450.2056 (127.43)	41,584.1245 (634.47)	809.8960 (762.26)	7;0	23.9299 (0.00)	24	1
test_R5_log[dataset1]	41,540.5000 (660.25)	227,430.2500 (>1000.0)	60,807.9184 (917.86)	39,994.5430 (>1000.0)	47,180.4585 (719.85)	15,654.4370 (>1000.0)	2;2	16.4452 (0.00)	24	1
test_R3_log[dataset15]	41,882.2090 (665.68)	45,651.8750 (429.83)	43,266.2569 (653.08)	904.7212 (256.07)	43,093.8330 (657.50)	1,241.2505 (>1000.0)	7;0	23.1127 (0.00)	24	1
test_R3_log[dataset16]	46,028.0420 (731.58)	76,241.6250 (717.85)	52,462.4067 (791.89)	9,163.3356 (>1000.0)	49,266.8330 (751.68)	7,375.9575 (>1000.0)	3;3	19.0613 (0.00)	21	1
test_R6_log[dataset1]	52,750.2920 (838.42)	53,230.7920 (501.19)	52,961.8575 (799.42)	142.3800 (40.30)	52,973.8330 (808.24)	233.3123 (219.59)	7;0	18.8815 (0.00)	19	1
test_R4_log[dataset2]	86,169.3330 (>1000.0)	89,802.1660 (845.53)	87,337.4512 (>1000.0)	1,183.8124 (335.06)	86,897.6250 (>1000.0)	1,971.4580 (>1000.0)	2;0	11.4498 (0.00)	12	1
test_R5_log[dataset2]	87,065.6670 (>1000.0)	95,364.7920 (897.91)	89,778.5681 (>1000.0)	2,574.2174 (728.60)	90,372.4170 (>1000.0)	3,450.5840 (>1000.0)	2;0	11.1385 (0.00)	11	1
test_R6_log[dataset2]	118,780.6250 (>1000.0)	119,480.9170 (>1000.0)	119,148.0001 (>1000.0)	256.7322 (72.67)	119,158.6670 (>1000.0)	457.6673 (430.75)	3;0	8.3929 (0.00)	9	1
test_R4_log[dataset3]	145,883.5830 (>1000.0)	152,703.7920 (>1000.0)	149,066.5953 (>1000.0)	2,663.1295 (753.77)	149,437.6670 (>1000.0)	4,586.6567 (>1000.0)	3;0	6.7084 (0.00)	7	1
test_R5_log[dataset3]	146,758.1250 (>1000.0)	198,395.8750 (>1000.0)	162,106.3333 (>1000.0)	22,865.0394 (>1000.0)	150,886.6670 (>1000.0)	34,534.3330 (>1000.0)	2;0	6.1688 (0.00)	7	1
test_R6_log[dataset3]	205,294.2920 (>1000.0)	209,015.2080 (>1000.0)	207,112.6834 (>1000.0)	1,804.8234 (510.83)	206,949.0840 (>1000.0)	3,544.1657 (>1000.0)	2;0	4.8283 (0.00)	5	1
test_R4_log[dataset4]	219,166.6670 (>1000.0)	223,362.8340 (>1000.0)	221,243.3420 (>1000.0)	1,975.4492 (559.13)	221,368.4170 (>1000.0)	3,809.8232 (>1000.0)	2;0	4.5199 (0.00)	5	1
test_R5_log[dataset4]	222,147.7920 (>1000.0)	261,772.5830 (>1000.0)	232,112.8668 (>1000.0)	16,793.1362 (>1000.0)	226,058.0420 (>1000.0)	14,572.2605 (>1000.0)	1;1	4.3082 (0.00)	5	1
test_R5_log[dataset5]	313,160.7920 (>1000.0)	314,962.0830 (>1000.0)	313,913.8668 (>1000.0)	775.6148 (219.53)	313,557.7500 (>1000.0)	1,279.9165 (>1000.0)	1;0	3.1856 (0.00)	5	1
test_R6_log[dataset4]	314,391.2500 (>1000.0)	317,650.3750 (>1000.0)	315,554.7916 (>1000.0)	1,307.9609 (370.20)	315,282.9170 (>1000.0)	1,766.9693 (>1000.0)	1;0	3.1690 (0.00)	5	1
test_R4_log[dataset5]	314,516.1250 (>1000.0)	340,992.9580 (>1000.0)	326,921.9834 (>1000.0)	12,016.4970 (>1000.0)	326,667.7090 (>1000.0)	22,515.9270 (>1000.0)	2;0	3.0588 (0.00)	5	1
test_R5_log[dataset6]	425,767.4170 (>1000.0)	436,244.7920 (>1000.0)	431,513.6086 (>1000.0)	4,119.1019 (>1000.0)	431,203.5840 (>1000.0)	6,214.8437 (>1000.0)	2;0	2.3174 (0.00)	5	1
test_R4_log[dataset6]	425,838.6660 (>1000.0)	433,285.1660 (>1000.0)	428,551.8748 (>1000.0)	3,030.2083 (857.66)	428,508.4170 (>1000.0)	4,235.1567 (>1000.0)	1;0	2.3334 (0.00)	5	1
test_R6_log[dataset5]	479,232.5000 (>1000.0)	483,818.6250 (>1000.0)	481,415.0916 (>1000.0)	1,932.2147 (546.89)	481,121.8750 (>1000.0)	3,347.1258 (>1000.0)	2;0	2.0772 (0.00)	5	1
test_R5_log[dataset7]	546,905.1670 (>1000.0)	588,077.5420 (>1000.0)	559,334.7166 (>1000.0)	17,036.0634 (>1000.0)	553,688.4580 (>1000.0)	20,438.5312 (>1000.0)	1;0	1.7878 (0.00)	5	1
test_R4_log[dataset7]	551,143.7500 (>1000.0)	569,702.9170 (>1000.0)	556,671.1336 (>1000.0)	7,533.6896 (>1000.0)	553,434.5420 (>1000.0)	7,458.8235 (>1000.0)	1;0	1.7964 (0.00)	5	1
test_R4_log[dataset8]	663,126.2500 (>1000.0)	694,416.4170 (>1000.0)	673,344.6168 (>1000.0)	12,926.9363 (>1000.0)	667,241.0830 (>1000.0)	16,788.2923 (>1000.0)	1;0	1.4851 (0.00)	5	1
test_R6_log[dataset6]	664,448.3330 (>1000.0)	817,269.7910 (>1000.0)	713,063.9332 (>1000.0)	60,717.6957 (>1000.0)	693,698.0420 (>1000.0)	61,757.6760 (>1000.0)	1;0	1.4024 (0.00)	5	1
test_R5_log[dataset8]	684,492.8330 (>1000.0)	746,958.0830 (>1000.0)	704,539.2998 (>1000.0)	25,127.9205 (>1000.0)	692,966.5000 (>1000.0)	27,811.1873 (>1000.0)	1;0	1.4194 (0.00)	5	1
test_R4_log[dataset9]	739,717.2080 (>1000.0)	744,539.2500 (>1000.0)	742,309.2752 (>1000.0)	2,060.8061 (583.29)	743,105.3750 (>1000.0)	3,472.4793 (>1000.0)	2;0	1.3471 (0.00)	5	1
test_R4_log[dataset10]	815,432.0420 (>1000.0)	891,119.0830 (>1000.0)	838,823.9414 (>1000.0)	30,047.3693 (>1000.0)	831,114.0410 (>1000.0)	26,067.9170 (>1000.0)	1;1	1.1921 (0.00)	5	1
test_R5_log[dataset9]	831,881.7500 (>1000.0)	841,217.7080 (>1000.0)	835,985.7584 (>1000.0)	3,410.8364 (965.40)	836,076.1250 (>1000.0)	3,662.4578 (>1000.0)	2;0	1.1962 (0.00)	5	1
test_R4_log[dataset11]	884,693.5000 (>1000.0)	1,054,632.7080 (>1000.0)	940,651.5168 (>1000.0)	73,639.5785 (>1000.0)	903,579.7920 (>1000.0)	109,351.3640 (>1000.0)	1;0	1.0631 (0.00)	5	1
test_R6_log[dataset7]	887,001.7920 (>1000.0)	1,191,276.0420 (>1000.0)	1,008,845.8500 (>1000.0)	124,005.5185 (>1000.0)	1,009,709.1250 (>1000.0)	189,781.8442 (>1000.0)	1;0	0.9912 (0.00)	5	1
test_R4_log[dataset12]	948,118.1250 (>1000.0)	1,037,137.8750 (>1000.0)	973,387.5416 (>1000.0)	36,782.7649 (>1000.0)	961,872.9580 (>1000.0)	37,737.0000 (>1000.0)	1;0	1.0273 (0.00)	5	1
test_R4_log[dataset13]	1,011,710.7080 (>1000.0)	1,582,500.3340 (>1000.0)	1,133,950.5834 (>1000.0)	251,017.2788 (>1000.0)	1,019,932.6670 (>1000.0)	162,967.0625 (>1000.0)	1;1	0.8819 (0.00)	5	1
test_R5_log[dataset10]	1,035,835.5420 (>1000.0)	1,519,345.3340 (>1000.0)	1,245,465.7504 (>1000.0)	210,554.0637 (>1000.0)	1,234,546.6670 (>1000.0)	373,741.8537 (>1000.0)	2;0	0.8029 (0.00)	5	1
test_R4_log[dataset14]	1,097,557.4590 (>1000.0)	1,607,511.9580 (>1000.0)	1,243,670.5334 (>1000.0)	212,386.1963 (>1000.0)	1,142,425.1250 (>1000.0)	231,774.7815 (>1000.0)	1;0	0.8041 (0.00)	5	1
test_R6_log[dataset8]	1,120,149.6250 (>1000.0)	1,132,784.0830 (>1000.0)	1,125,021.4666 (>1000.0)	4,741.5529 (>1000.0)	1,123,359.0000 (>1000.0)	4,807.9583 (>1000.0)	2;0	0.8889 (0.00)	5	1
test_R4_log[dataset15]	1,179,233.7920 (>1000.0)	1,661,220.7500 (>1000.0)	1,298,479.7252 (>1000.0)	203,581.6132 (>1000.0)	1,220,122.5840 (>1000.0)	135,725.1145 (>1000.0)	1;1	0.7701 (0.00)	5	1
test_R5_log[dataset11]	1,207,249.8750 (>1000.0)	1,488,714.2080 (>1000.0)	1,268,452.3832 (>1000.0)	123,199.7640 (>1000.0)	1,215,676.4160 (>1000.0)	74,907.2710 (>1000.0)	1;1	0.7884 (0.00)	5	1
test_R4_log[dataset16]	1,227,012.2920 (>1000.0)	1,758,462.9160 (>1000.0)	1,474,081.4998 (>1000.0)	255,771.0387 (>1000.0)	1,336,533.2500 (>1000.0)	461,181.3743 (>1000.0)	2;0	0.6784 (0.00)	5	1
test_R6_log[dataset9]	1,363,283.8750 (>1000.0)	1,530,419.6660 (>1000.0)	1,400,079.6166 (>1000.0)	72,919.9715 (>1000.0)	1,368,643.6250 (>1000.0)	45,119.6980 (>1000.0)	1;1	0.7142 (0.00)	5	1
test_R5_log[dataset12]	1,422,237.4590 (>1000.0)	1,788,747.7080 (>1000.0)	1,534,032.8168 (>1000.0)	159,556.5382 (>1000.0)	1,434,631.4170 (>1000.0)	215,446.7497 (>1000.0)	1;0	0.6519 (0.00)	5	1
test_R5_log[dataset13]	1,650,810.0000 (>1000.0)	1,971,062.0000 (>1000.0)	1,754,116.9084 (>1000.0)	124,850.3719 (>1000.0)	1,715,724.2920 (>1000.0)	98,678.4380 (>1000.0)	1;1	0.5701 (0.00)	5	1
test_R6_log[dataset10]	1,713,193.4590 (>1000.0)	1,758,007.9580 (>1000.0)	1,726,207.1418 (>1000.0)	18,054.5780 (>1000.0)	1,719,123.7500 (>1000.0)	13,468.5932 (>1000.0)	1;1	0.5793 (0.00)	5	1
test_R5_log[dataset14]	1,906,932.5000 (>1000.0)	2,241,791.0420 (>1000.0)	2,030,156.3752 (>1000.0)	163,350.1442 (>1000.0)	1,916,431.2920 (>1000.0)	277,864.4790 (>1000.0)	1;0	0.4926 (0.00)	5	1
test_R6_log[dataset11]	2,042,211.1670 (>1000.0)	2,097,642.0840 (>1000.0)	2,062,791.3754 (>1000.0)	22,973.2420 (>1000.0)	2,059,277.4170 (>1000.0)	35,144.0725 (>1000.0)	1;0	0.4848 (0.00)	5	1
test_R5_log[dataset15]	2,163,320.2920 (>1000.0)	2,446,427.0410 (>1000.0)	2,281,980.1916 (>1000.0)	148,952.2620 (>1000.0)	2,188,543.7500 (>1000.0)	276,879.5305 (>1000.0)	2;0	0.4382 (0.00)	5	1
test_R6_log[dataset12]	2,370,314.7920 (>1000.0)	2,438,003.0000 (>1000.0)	2,403,924.8416 (>1000.0)	25,512.7316 (>1000.0)	2,407,873.0410 (>1000.0)	34,900.2082 (>1000.0)	2;0	0.4160 (0.00)	5	1
test_R5_log[dataset16]	2,463,318.4580 (>1000.0)	3,142,776.6670 (>1000.0)	2,735,187.6916 (>1000.0)	283,950.6552 (>1000.0)	2,609,124.8750 (>1000.0)	443,448.6463 (>1000.0)	1;0	0.3656 (0.00)	5	1
test_R6_log[dataset13]	2,779,916.5410 (>1000.0)	3,255,206.9170 (>1000.0)	2,951,596.9834 (>1000.0)	216,075.9480 (>1000.0)	2,828,930.1250 (>1000.0)	356,021.9695 (>1000.0)	1;0	0.3388 (0.00)	5	1
test_R6_log[dataset14]	3,231,256.5830 (>1000.0)	3,386,528.6660 (>1000.0)	3,290,461.7914 (>1000.0)	57,944.0951 (>1000.0)	3,273,249.6670 (>1000.0)	54,065.8020 (>1000.0)	2;0	0.3039 (0.00)	5	1
test_R6_log[dataset15]	3,692,611.0420 (>1000.0)	3,787,663.7500 (>1000.0)	3,755,966.0834 (>1000.0)	39,184.9971 (>1000.0)	3,777,172.4160 (>1000.0)	50,150.6453 (>1000.0)	1;0	0.2662 (0.00)	5	1
test_R6_log[dataset16]	4,193,485.1660 (>1000.0)	4,384,859.5410 (>1000.0)	4,270,239.2996 (>1000.0)	72,261.4185 (>1000.0)	4,266,918.6250 (>1000.0)	85,345.5932 (>1000.0)	2;0	0.2342 (0.00)	5	1

Fig. 1. pytest-benchmark results for LOG log [2].

benchmark: 60 tests

Name (time in us)	Min	Max	Mean	StdDev	Median	IQR	Outliers	OPS	Rounds	Iterations
test_R1_om[dataset0]	150.8340 (1.0)	256.5000 (1.0)	156.1299 (1.0)	6.8289 (1.0)	154.1670 (1.0)	3.2090 (1.0)	230;284	6,404.9238 (1.0)	2333	1
test_R1_om[dataset1]	308.3330 (2.04)	11,961.5840 (46.63)	396.6170 (2.54)	670.7994 (98.23)	316.2080 (2.05)	19.6975 (6.14)	17;124	2,521.3241 (0.39)	1065	1
test_R1_om[dataset2]	471.2080 (3.12)	773.7500 (3.02)	493.8532 (3.16)	35.8624 (5.25)	484.1875 (3.14)	10.5840 (3.30)	19;42	2,024.8931 (0.32)	354	1
test_R1_om[dataset3]	650.7920 (4.31)	1,026.4160 (4.00)	675.2195 (4.32)	39.1981 (5.74)	664.4580 (4.31)	13.8852 (4.33)	40;64	1,481.0000 (0.23)	545	1
test_R1_om[dataset4]	840.9580 (5.58)	1,262.4580 (4.92)	861.8095 (5.52)	30.9670 (4.53)	856.3540 (5.55)	10.7085 (3.34)	21;35	1,160.3493 (0.18)	424	1
test_R1_om[dataset5]	1,021.7090 (6.77)	1,474.8330 (5.75)	1,070.8693 (6.86)	62.6390 (9.17)	1,046.9580 (6.79)	37.5207 (11.69)	36;38	933.8208 (0.15)	349	1
test_R1_om[dataset6]	1,224.0420 (8.12)	1,751.5000 (6.83)	1,259.0407 (8.06)	51.4102 (7.53)	1,244.7495 (8.07)	20.4585 (6.38)	29;36	794.2555 (0.12)	308	1
test_R1_om[dataset7]	1,410.9580 (9.35)	2,388.5830 (9.31)	1,467.1144 (9.40)	82.8794 (12.14)	1,444.7920 (9.37)	22.7500 (7.09)	20;33	681.6101 (0.11)	263	1
test_R1_om[dataset8]	1,630.1670 (10.81)	2,330.0000 (9.08)	1,717.2561 (11.00)	101.3516 (14.84)	1,673.2080 (10.85)	80.4790 (25.08)	29;21	582.3243 (0.09)	217	1
test_R1_om[dataset9]	1,845.4170 (12.23)	2,725.1670 (10.62)	2,001.1744 (12.82)	177.9771 (26.06)	1,920.3330 (12.46)	154.4995 (48.15)	15;8	499.7066 (0.08)	97	1
test_R4_om[dataset0]	36,534.0000 (242.21)	38,192.4170 (148.90)	36,788.4568 (235.63)	366.0104 (53.60)	36,701.8340 (238.07)	170.5630 (53.15)	2;2	27.1824 (0.00)	27	1
test_R5_om[dataset0]	36,577.5000 (242.50)	38,663.0420 (150.73)	36,839.1852 (235.95)	438.6304 (64.23)	36,723.2080 (238.20)	155.7713 (48.54)	2;2	27.1450 (0.00)	27	1
test_R2_om[dataset0]	39,385.2500 (261.12)	44,129.2500 (172.04)	40,187.2309 (257.40)	1,074.1223 (157.29)	39,665.4790 (257.29)	1,148.6460 (357.94)	3;1	24.8835 (0.00)	24	1
test_R6_om[dataset0]	39,408.4160 (261.27)	70,777.2500 (275.93)	41,928.4666 (268.55)	6,569.1365 (961.97)	39,650.4580 (257.19)	1,254.6355 (390.97)	2;4	23.8501 (0.00)	25	1
test_R3_om[dataset0]	39,676.1250 (263.04)	42,360.7500 (165.15)	40,253.2048 (257.82)	568.5502 (83.26)	40,145.3750 (260.40)	379.8437 (118.37)	5;2	24.8427 (0.00)	25	1
test_R5_om[dataset1]	108,321.1660 (718.15)	109,130.8750 (425.46)	108,733.7374 (696.43)	288.8875 (42.30)	108,676.0830 (704.92)	559.1250 (174.24)	4;0	9.1968 (0.00)	10	1
test_R4_om[dataset1]	108,529.8750 (719.53)	109,049.2500 (425.14)	108,727.7086 (696.39)	174.4373 (25.54)	108,666.9380 (704.87)	205.3330 (63.99)	3;0	9.1973 (0.00)	10	1
test_R2_om[dataset1]	109,740.0830 (727.56)	110,821.8750 (432.05)	110,091.2407 (705.13)	345.7146 (50.63)	109,983.0830 (713.40)	305.3438 (95.15)	3;1	9.0834 (0.00)	9	1
test_R6_om[dataset1]	109,822.6250 (728.10)	123,754.0000 (482.47)	111,987.8646 (717.27)	4,780.0086 (699.97)	110,170.2915 (714.62)	1,050.5215 (327.37)	1;1	8.9295 (0.00)	8	1
test_R3_om[dataset1]	110,950.0420 (735.58)	136,333.5410 (531.51)	118,230.5323 (757.26)	9,908.0806 (>1000.0)	111,519.4170 (723.37)	13,522.1882 (>1000.0)	2;0	8.4581 (0.00)	9	1
test_R4_om[dataset2]	213,475.5830 (>1000.0)	214,781.0840 (837.35)	214,186.6582 (>1000.0)	542.9613 (79.51)	214,201.9580 (>1000.0)	921.5633 (287.18)	2;0	4.6688 (0.00)	5	1
test_R5_om[dataset2]	213,591.1660 (>1000.0)	220,893.6250 (861.18)	215,488.0916 (>1000.0)	3,042.5945 (445.55)	214,370.5420 (>1000.0)	2,167.5210 (675.45)	1;1	4.6406 (0.00)	5	1
test_R6_om[dataset2]	213,996.2500 (>1000.0)	221,040.5420 (861.76)	216,307.1584 (>1000.0)	2,889.8959 (423.19)	215,547.3330 (>1000.0)	3,801.6348 (>1000.0)	1;0	4.6231 (0.00)	5	1
test_R2_om[dataset2]	214,539.5830 (>1000.0)	215,942.2080 (841.88)	214,932.5582 (>1000.0)	572.1833 (83.79)	214,761.2080 (>1000.0)	440.8752 (137.39)	1;1	4.6526 (0.00)	5	1
test_R3_om[dataset2]	217,387.9170 (>1000.0)	295,456.0000 (>1000.0)	233,390.5084 (>1000.0)	34,697.5106 (>1000.0)	218,018.3750 (>1000.0)	19,987.4582 (>1000.0)	1;1	4.2847 (0.00)	5	1
test_R6_om[dataset3]	359,530.0830 (>1000.0)	373,441.3330 (>1000.0)	363,408.6998 (>1000.0)	5,690.6586 (833.33)	361,147.7500 (>1000.0)	4,613.8128 (>1000.0)	1;1	2.7517 (0.00)	5	1
test_R2_om[dataset3]	360,257.4580 (>1000.0)	361,834.7920 (>1000.0)	360,808.8166 (>1000.0)	607.9581 (89.03)	360,738.5420 (>1000.0)	593.1773 (184.85)	1;0	2.7716 (0.00)	5	1
test_R3_om[dataset3]	361,293.9590 (>1000.0)	369,585.0420 (>1000.0)	364,260.7752 (>1000.0)	3,151.7279 (461.53)	363,372.6250 (>1000.0)	2,969.9583 (925.50)	1;0	2.7453 (0.00)	5	1
test_R4_om[dataset3]	365,751.4170 (>1000.0)	367,238.6670 (>1000.0)	366,124.6918 (>1000.0)	627.2512 (91.85)	365,910.7920 (>1000.0)	472.3747 (147.20)	1;1	2.7313 (0.00)	5	1
test_R5_om[dataset3]	367,321.2920 (>1000.0)	368,573.8750 (>1000.0)	367,913.8418 (>1000.0)	530.2182 (77.64)	367,882.5410 (>1000.0)	928.0835 (289.21)	2;0	2.7180 (0.00)	5	1
test_R2_om[dataset4]	517,430.0420 (>1000.0)	530,027.0000 (>1000.0)	522,605.2668 (>1000.0)	6,553.9666 (959.75)	518,466.7090 (>1000.0)	12,088.7393 (>1000.0)	2;0	1.9135 (0.00)	5	1
test_R3_om[dataset4]	517,866.3750 (>1000.0)	580,775.5830 (>1000.0)	542,757.9250 (>1000.0)	23,890.3259 (>1000.0)	543,286.1670 (>1000.0)	27,564.9270 (>1000.0)	2;0	1.8424 (0.00)	5	1
test_R6_om[dataset4]	518,761.7080 (>1000.0)	545,910.0410 (>1000.0)	526,459.8082 (>1000.0)	11,600.4860 (>1000.0)	520,262.7500 (>1000.0)	14,049.3018 (>1000.0)	1;0	1.8995 (0.00)	5	1
test_R5_om[dataset4]	532,664.4580 (>1000.0)	537,153.0000 (>1000.0)	535,366.9582 (>1000.0)	1,994.5994 (292.08)	536,380.7080 (>1000.0)	3,343.9168 (>1000.0)	1;0	1.8679 (0.00)	5	1
test_R4_om[dataset4]	533,521.0000 (>1000.0)	536,265.2080 (>1000.0)	535,257.2834 (>1000.0)	1,181.5274 (173.02)	535,648.0000 (>1000.0)	1,915.5833 (596.94)	1;0	1.8683 (0.00)	5	1
test_R2_om[dataset5]	613,859.6250 (>1000.0)	618,429.5420 (>1000.0)	615,348.5086 (>1000.0)	1,908.2027 (279.43)	614,729.8750 (>1000.0)	2,642.0420 (823.32)	1;0	1.6251 (0.00)	5	1
test_R4_om[dataset5]	639,048.6670 (>1000.0)	664,370.5840 (>1000.0)	644,784.7586 (>1000.0)	10,978.1569 (>1000.0)	639,699.2920 (>1000.0)	7,541.7923 (>1000.0)	1;1	1.5509 (0.00)	5	1
test_R2_om[dataset6]	698,532.2090 (>1000.0)	701,486.0420 (>1000.0)	699,600.0002 (>1000.0)	1,199.1265 (175.60)	699,350.7920 (>1000.0)	1,704.9263 (531.29)	1;0	1.4294 (0.00)	5	1
test_R4_om[dataset6]	728,548.4590 (>1000.0)	738,092.2500 (>1000.0)	731,199.6168 (>1000.0)	3,936.5899 (576.46)	729,927.1250 (>1000.0)	3,651.3223 (>1000.0)	1;1	1.3676 (0.00)	5	1
test_R6_om[dataset5]	728,918.7090 (>1000.0)	751,684.6660 (>1000.0)	736,043.0250 (>1000.0)	9,098.4061 (>1000.0)	734,302.8330 (>1000.0)	8,997.3010 (>1000.0)	1;0	1.3586 (0.00)	5	1
test_R3_om[dataset5]	740,734.5830 (>1000.0)	758,582.2500 (>1000.0)	746,894.5332 (>1000.0)	7,774.5074 (>1000.0)	742,988.9170 (>1000.0)	12,010.8543 (>1000.0)	1;0	1.3389 (0.00)	5	1
test_R5_om[dataset5]	753,760.7090 (>1000.0)	756,551.5410 (>1000.0)	755,046.1166 (>1000.0)	1,293.4334 (189.41)	754,968.3330 (>1000.0)	2,458.8955 (766.25)	1;0	1.3244 (0.00)	5	1
test_R2_om[dataset7]	786,544.2500 (>1000.0)	795,611.2920 (>1000.0)	790,796.9334 (>1000.0)	4,243.5271 (621.41)	789,614.7080 (>1000.0)	7,995.9483 (>1000.0)	2;0	1.2645 (0.00)	5	1
test_R4_om[dataset7]	822,238.9160 (>1000.0)	829,896.5830 (>1000.0)	825,222.4496 (>1000.0)	2,927.2959 (428.67)	824,952.5830 (>1000.0)	3,572.9788 (>1000.0)	2;0	1.2118 (0.00)	5	1
test_R2_om[dataset8]	876,114.2910 (>1000.0)	912,592.4590 (>1000.0)	890,702.4918 (>1000.0)	19,643.1049 (>1000.0)	876,574.7090 (>1000.0)	35,710.7920 (>1000.0)	2;0	1.1227 (0.00)	5	1
test_R4_om[dataset8]	915,587.3340 (>1000.0)	916,341.6250 (>1000.0)	915,942.3834 (>1000.0)	297.7379 (43.60)	915,935.0830 (>1000.0)	465.9790 (145.21)	2;0	1.0918 (0.00)	5	1
test_R6_om[dataset6]	964,984.3750 (>1000.0)	983,822.3330 (>1000.0)	970,262.2916 (>1000.0)	7,836.2622 (>1000.0)	966,357.2500 (>1000.0)	7,903.4890 (>1000.0)	1;0	1.0306 (0.00)	5	1
test_R2_om[dataset9]	972,126.9580 (>1000.0)	1,155,252.9590 (>1000.0)	1,016,054.5168 (>1000.0)	78,447.0473 (>1000.0)	979,549.4590 (>1000.0)	62,571.7817 (>1000.0)	1;1	0.9842 (0.00)	5	1
test_R3_om[dataset6]	979,649.2080 (>1000.0)	1,044,501.7080 (>1000.0)	1,003,358.4834 (>1000.0)	26,559.3599 (>1000.0)	1,005,412.6670 (>1000.0)	36,686.6250 (>1000.0)	1;0	0.9967 (0.00)	5	1
test_R5_om[dataset6]	999,730.3750 (>1000.0)	1,005,028.0000 (>1000.0)	1,001,325.6998 (>1000.0)	2,154.3239 (315.47)	1,000,774.9580 (>1000.0)	2,271.9683 (707.99)	1;0	0.9987 (0.00)	5	1
test_R4_om[dataset9]	1,014,443.1250 (>1000.0)	1,017,759.4580 (>1000.0)	1,015,812.2000 (>1000.0)	1,286.2287 (188.35)	1,015,593.9170 (>1000.0)	1,800.5838 (561.10)	2;0	0.9844 (0.00)	5	1
test_R6_om[dataset7]	1,246,129.5410 (>1000.0)	1,288,086.6250 (>1000.0)	1,260,953.1332 (>1000.0)	19,903.5095 (>1000.0)	1,247,363.5830 (>1000.0)	32,817.0837 (>1000.0)	1;0	0.7931 (0.00)	5	1
test_R3_om[dataset7]	1,264,482.4170 (>1000.0)	1,302,578.0830 (>1000.0)	1,283,925.3748 (>1000.0)	14,967.5848 (>1000.0)	1,280,025.5410 (>1000.0)	22,249.8542 (>1000.0)	2;0	0.7789 (0.00)	5	1
test_R5_om[dataset7]	1,294,315.5830 (>1000.0)	1,514,919.6250 (>1000.0)	1,340,006.0416 (>1000.0)	97,792.9472 (>1000.0)	1,297,161.2500 (>1000.0)	57,564.5720 (>1000.0)	1;1	0.7463 (0.00)	5	1
test_R6_om[dataset8]	1,576,857.7090 (>1000.0)	1,632,523.0000 (>1000.0)	1,593,270.3416 (>1000.0)	23,299.4694 (>1000.0)	1,580,733.3750 (>1000.0)	26,952.6668 (>1000.0)	1;0	0.6276 (0.00)	5	1
test_R3_om[dataset8]	1,618,072.2500 (>1000.0)	1,750,549.6250 (>1000.0)	1,670,476.9750 (>1000.0)	51,188.9952 (>1000.0)	1,657,002.4170 (>1000.0)	67,342.3440 (>1000.0)	2;0	0.5986 (0.00)	5	1
test_R5_om[dataset8]	1,634,780.6250 (>1000.0)	1,794,320.4590 (>1000.0)	1,667,391.2500 (>1000.0)	70,958.5717 (>1000.0)	1,635,929.2080 (>1000.0)	40,750.6145 (>1000.0)	1;1	0.5997 (0.00)	5	1
test_R6_om[dataset9]	1,942,125.9160 (>1000.0)	1,952,235.6250 (>1000.0)	1,947,911.9080 (>1000.0)	4,136.0345 (605.67)	1,948,442.2500 (>1000.0)	6,708.5520 (>1000.0)	2;0	0.5134 (0.00)	5	1
test_R3_om[dataset9]	1,944,067.6670 (>1000.0)	1,980,257.4580 (>1000.0)	1,959,812.9668 (>1000.0)	14,591.6759 (>1000.0)	1,956,575.6670 (>1000.0)	23,047.6662 (>1000.0)	2;0	0.5103 (0.00)	5	1
test_R5_om[dataset9]	2,023,492.3750 (>1000.0)	2,082,712.6670 (>1000.0)	2,041,996.5002 (>1000.0)	23,879.5135 (>1000.0)	2,032,801.4170 (>1000.0)	26,002.2295 (>1000.0)	1;0	0.4897 (0.00)	5	1

Fig. 2. `pytest-benchmark` results for OM log [1].

benchmark: 35 tests

Name (time in us)	Min	Max	Mean	StdDev	Median	IQR	Outliers	OPS	Rounds	Iterations
test_R1_p2p[dataset0]	161.0410 (1.0)	468.6670 (1.0)	167.6511 (1.0)	10.1984 (1.0)	165.5840 (1.0)	4.0630 (1.0)	196;307	5,964.7672 (1.0)	2476	1
test_R1_p2p[dataset1]	333.4170 (2.07)	1,265.2920 (2.70)	350.9369 (2.09)	35.9020 (3.52)	343.2500 (2.07)	14.6147 (3.60)	47;69	2,849.5155 (0.48)	1327	1
test_R1_p2p[dataset2]	510.6250 (3.17)	20,003.0840 (42.68)	943.0371 (5.62)	1,852.3808 (181.63)	543.2500 (3.28)	99.1048 (24.39)	42;147	1,060.4037 (0.18)	1039	1
test_R1_p2p[dataset3]	680.6660 (4.23)	1,593.5000 (3.40)	725.4461 (4.33)	69.3485 (6.80)	704.1250 (4.25)	37.7612 (9.29)	64;69	1,378.4622 (0.23)	729	1
test_R1_p2p[dataset4]	847.8750 (5.26)	1,125.1660 (2.40)	874.6368 (5.22)	26.3123 (2.58)	867.7080 (5.24)	15.3755 (3.78)	63;63	1,143.3317 (0.19)	567	1
test_R1_p2p[dataset5]	1,034.3330 (6.42)	1,347.3750 (2.87)	1,064.2802 (6.35)	35.0949 (3.44)	1,056.0000 (6.38)	17.0733 (4.20)	38;49	939.6022 (0.16)	499	1
test_R1_p2p[dataset6]	1,214.4590 (7.54)	1,873.2500 (4.00)	1,269.3230 (7.57)	76.2136 (7.47)	1,241.5420 (7.50)	47.6047 (11.72)	39;36	787.8215 (0.13)	415	1
test_R3_p2p[dataset0]	35,613.2920 (221.14)	75,442.1660 (160.97)	38,398.8108 (229.04)	7,669.7195 (752.05)	36,120.6040 (218.14)	2,606.0000 (641.40)	1;1	26.0425 (0.00)	26	1
test_R2_p2p[dataset0]	35,692.2080 (221.63)	55,061.6250 (117.49)	38,436.7500 (229.27)	4,502.1743 (441.46)	36,439.0000 (220.06)	3,075.8440 (757.04)	4;4	26.0168 (0.00)	27	1
test_R4_p2p[dataset0]	64,900.5420 (403.01)	66,806.7920 (142.55)	65,289.9038 (389.44)	452.9698 (44.42)	65,203.3125 (393.78)	379.5205 (93.41)	1;1	15.3163 (0.00)	16	1
test_R5_p2p[dataset0]	65,217.5000 (404.97)	69,301.5420 (147.87)	66,034.4076 (393.88)	1,025.5711 (100.56)	65,777.5830 (397.25)	691.0420 (170.08)	1;1	15.1436 (0.00)	14	1
test_R3_p2p[dataset1]	141,702.4170 (879.92)	142,369.7500 (303.78)	142,005.0537 (847.03)	219.1310 (21.49)	142,024.6670 (857.72)	276.5105 (68.06)	2;0	7.0420 (0.00)	7	1
test_R2_p2p[dataset1]	141,879.5830 (881.02)	147,945.8330 (315.67)	143,738.0534 (857.36)	2,555.9980 (250.63)	142,506.4170 (860.63)	3,654.7808 (899.53)	2;0	6.9571 (0.00)	7	1
test_R5_p2p[dataset1]	256,653.0000 (>1000.0)	262,890.2500 (560.93)	259,696.3168 (>1000.0)	2,646.1699 (259.47)	260,492.4170 (>1000.0)	4,457.0628 (>1000.0)	2;0	3.8507 (0.00)	5	1
test_R4_p2p[dataset1]	257,924.4580 (>1000.0)	266,498.6670 (568.63)	263,255.9000 (>1000.0)	3,354.2011 (328.89)	263,993.2920 (>1000.0)	4,419.9275 (>1000.0)	1;0	3.7986 (0.00)	5	1
test_R3_p2p[dataset2]	320,312.2080 (>1000.0)	322,318.6670 (687.73)	321,143.0082 (>1000.0)	841.8462 (82.55)	320,818.7910 (>1000.0)	1,361.3960 (335.07)	1;0	3.1139 (0.00)	5	1
test_R2_p2p[dataset2]	321,114.6250 (>1000.0)	335,427.7500 (715.71)	327,540.8248 (>1000.0)	6,735.7743 (660.47)	325,795.8330 (>1000.0)	12,757.5943 (>1000.0)	1;0	3.0531 (0.00)	5	1
test_R2_p2p[dataset3]	518,285.2910 (>1000.0)	704,161.1670 (>1000.0)	565,506.0832 (>1000.0)	79,388.7572 (>1000.0)	526,824.0830 (>1000.0)	77,710.1253 (>1000.0)	1;0	1.7683 (0.00)	5	1
test_R3_p2p[dataset3]	573,026.2500 (>1000.0)	618,252.7500 (>1000.0)	590,494.9334 (>1000.0)	22,497.8276 (>1000.0)	576,001.0000 (>1000.0)	39,964.9372 (>1000.0)	1;0	1.6935 (0.00)	5	1
test_R5_p2p[dataset2]	575,972.7080 (>1000.0)	592,988.0410 (>1000.0)	584,566.1748 (>1000.0)	7,686.8295 (753.73)	588,320.0420 (>1000.0)	13,195.2710 (>1000.0)	3;0	1.7107 (0.00)	5	1
test_R4_p2p[dataset2]	580,385.0420 (>1000.0)	763,018.5000 (>1000.0)	639,042.7168 (>1000.0)	75,860.9082 (>1000.0)	598,176.3340 (>1000.0)	95,855.5825 (>1000.0)	1;0	1.5648 (0.00)	5	1
test_R2_p2p[dataset4]	655,910.5830 (>1000.0)	781,538.5830 (>1000.0)	687,265.5082 (>1000.0)	53,237.2818 (>1000.0)	666,709.5000 (>1000.0)	44,243.3437 (>1000.0)	1;1	1.4550 (0.00)	5	1
test_R2_p2p[dataset5]	814,126.7500 (>1000.0)	907,829.6670 (>1000.0)	851,281.5000 (>1000.0)	39,187.7354 (>1000.0)	841,486.5830 (>1000.0)	63,430.1667 (>1000.0)	1;0	1.1747 (0.00)	5	1
test_R3_p2p[dataset4]	905,571.4170 (>1000.0)	911,908.6250 (>1000.0)	908,807.0998 (>1000.0)	2,279.6937 (223.53)	908,959.0830 (>1000.0)	2,456.1455 (604.51)	2;0	1.1003 (0.00)	5	1
test_R4_p2p[dataset3]	918,334.0000 (>1000.0)	923,346.3750 (>1000.0)	920,719.5416 (>1000.0)	2,129.0379 (208.76)	919,703.7920 (>1000.0)	3,429.0005 (843.96)	2;0	1.0861 (0.00)	5	1
test_R5_p2p[dataset3]	1,026,855.7080 (>1000.0)	1,032,024.0830 (>1000.0)	1,029,373.9498 (>1000.0)	2,239.4330 (219.59)	1,030,056.3330 (>1000.0)	3,868.0625 (952.02)	2;0	0.9715 (0.00)	5	1
test_R2_p2p[dataset6]	1,056,263.2080 (>1000.0)	1,977,077.6670 (>1000.0)	1,290,535.5250 (>1000.0)	388,573.1519 (>1000.0)	1,179,618.7500 (>1000.0)	320,011.3028 (>1000.0)	1;1	0.7749 (0.00)	5	1
test_R4_p2p[dataset4]	1,146,054.5410 (>1000.0)	1,166,236.6670 (>1000.0)	1,153,525.8750 (>1000.0)	9,030.0425 (885.44)	1,148,275.3750 (>1000.0)	14,604.0625 (>1000.0)	1;0	0.8669 (0.00)	5	1
test_R3_p2p[dataset5]	1,370,573.3750 (>1000.0)	1,560,289.7090 (>1000.0)	1,416,713.3002 (>1000.0)	80,740.6612 (>1000.0)	1,381,801.0000 (>1000.0)	60,830.1458 (>1000.0)	1;1	0.7059 (0.00)	5	1
test_R4_p2p[dataset5]	1,410,134.2500 (>1000.0)	1,479,196.8750 (>1000.0)	1,425,067.8084 (>1000.0)	30,299.8995 (>1000.0)	1,411,012.4590 (>1000.0)	19,820.4998 (>1000.0)	1;1	0.7017 (0.00)	5	1
test_R4_p2p[dataset6]	1,683,241.2910 (>1000.0)	1,708,478.0000 (>1000.0)	1,692,950.5748 (>1000.0)	10,102.0850 (990.55)	1,692,365.6250 (>1000.0)	14,442.5838 (>1000.0)	1;0	0.5907 (0.00)	5	1
test_R5_p2p[dataset4]	1,753,207.5000 (>1000.0)	1,948,250.3750 (>1000.0)	1,819,123.1502 (>1000.0)	77,960.0355 (>1000.0)	1,798,717.0420 (>1000.0)	95,715.6567 (>1000.0)	1;0	0.5497 (0.00)	5	1
test_R3_p2p[dataset6]	1,926,816.9580 (>1000.0)	2,125,688.4580 (>1000.0)	2,018,692.0080 (>1000.0)	89,170.7291 (>1000.0)	2,012,635.2910 (>1000.0)	165,123.5633 (>1000.0)	2;0	0.4954 (0.00)	5	1
test_R5_p2p[dataset5]	2,426,131.7500 (>1000.0)	3,150,964.4170 (>1000.0)	2,683,030.7500 (>1000.0)	313,361.9195 (>1000.0)	2,527,205.0830 (>1000.0)	484,968.5407 (>1000.0)	1;0	0.3727 (0.00)	5	1
test_R5_p2p[dataset6]	3,400,816.2500 (>1000.0)	3,414,444.5410 (>1000.0)	3,408,074.1082 (>1000.0)	5,136.4153 (503.65)	3,409,600.8750 (>1000.0)	6,716.7290 (>1000.0)	2;0	0.2934 (0.00)	5	1

Fig. 3. `pytest-benchmark` results for P2P log [3].

Chunk	**Events**	**R1**	**R2**	**R3**	**R4**	**R5**	R6
chunk1	2000	0.00014	0.02605	0.02744	0.04009	0.03998	0.02846
chunk2	4000	0.00033	0.09024	0.09025	0.14348	0.14045	0.08872
chunk3	6000	0.00046	0.18266	0.36661	0.30223	0.35947	0.17751
chunk4	8000	0.00057	0.43306	0.35323	0.49675	0.53383	0.30676
chunk5	10000	0.00073	0.46347	0.50922	0.65759	0.82230	0.44925
chunk6	12000	0.00088	0.51812	0.74641	0.86764	1.19508	0.67119
chunk7	14000	0.00105	0.64941	1.04399	1.00240	1.72030	0.89050

Table 1. Formatted results for Figure 4 graphs building.

Chunk	**Events**	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**
chunk1	10000	0.00011	0.00167	0.00159	0.03122	0.03229	0.04093
chunk2	20000	0.00022	0.00419	0.00442	0.11890	0.12400	0.17098
chunk3	30000	0.00034	0.00725	0.00756	0.31860	0.26053	0.38631
chunk4	40000	0.00046	0.01117	0.01109	0.44774	0.47895	0.74046
chunk5	50000	0.00059	0.01572	0.01554	0.69382	0.74019	1.11503
chunk6	60000	0.00073	0.01809	0.02748	0.92580	1.11508	1.71172
chunk7	70000	0.00092	0.02551	0.02796	1.08719	1.46587	2.37199
chunk8	80000	0.00116	0.03780	0.03615	1.14796	1.85143	3.37935
chunk9	90000	0.00199	0.02761	0.04452	1.27740	2.46438	4.35435
chunk10	100000	0.00224	0.03247	0.06201	1.45969	3.30174	5.57581

Table 2. Formatted results for Figure 5 graphs building.

tables present the key metrics used for the final performance analysis and visualization. They consolidate the execution times obtained for each test scenario, enabling a comparative assessment of the technique's behavior across different event logs.

The processed results serve as the basis for generating graphical representations that illustrate performance trends. These visualizations are designed to highlight execution time variations, efficiency differences between different test cases, and potential performance bottlenecks. By analyzing these trends, we gain insights into the computational cost of applying OCCRs of increasing complexity, i.e., from **R1** to **R6**, under different conditions represented by the increasing event log chunks size, i.e., from **chunk1** to **chunk7** in Table 1, or to **chunk10** in Table 2.

The final tables obtained through results processing are used to generate the performance analysis graphs included in the "Performance Analysis" section of the paper and reported here in Figures 4 and 5. These graphs provide a visual representation of execution time distributions offering a clear interpretation of the technique's scalability and computational demands.

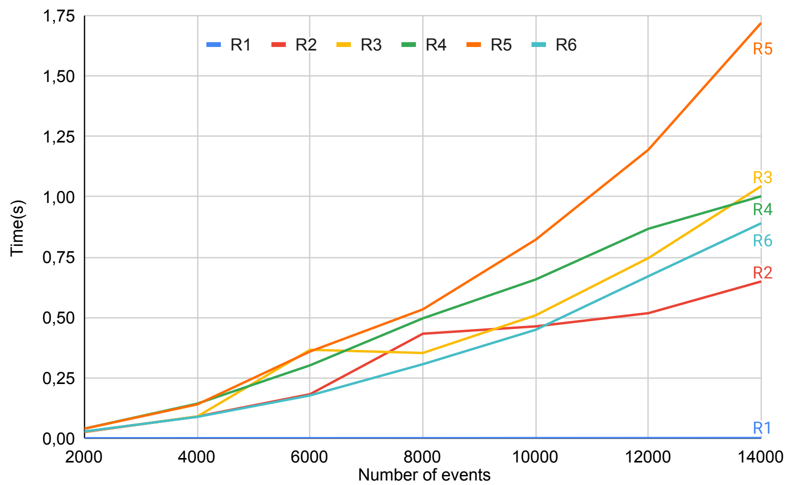

Fig. 4. Results of the performance analysis on different increasing object-centric event logs.

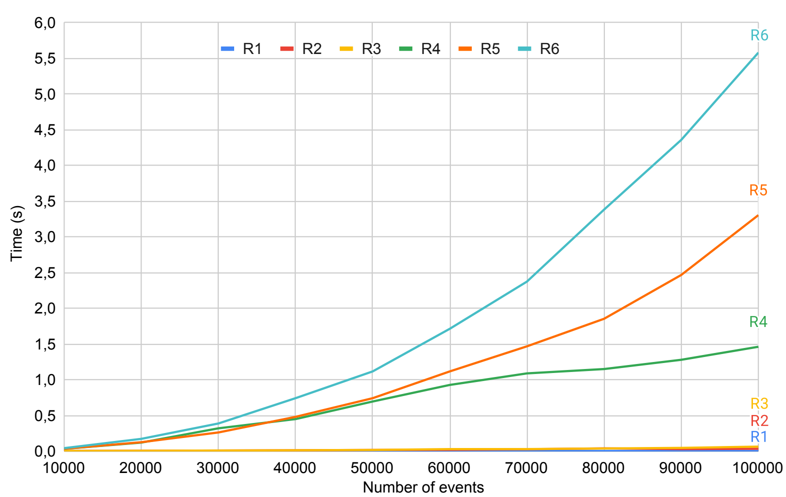

Fig. 5. Results of the performance analysis on widened logistics object-centric event log [2].

2 Technique Applicability

To demonstrate the applicability of the technique across different domains, we conducted some demonstrations on the synthetic event logs [2,1,3], applying object-centric compliance rules specific to their respective domains.

2.1 Preliminaries

Below are three case studies proposed by the community, along with the Compliance Requirements (CRs) that illustrate the applicability of our Object-Centric Compliance Checking technique. All the cases are simulated object-centric event logs.

Logistics (LOG) [2]. Captures logistics operations. The logistics process involves the movement of goods from the production site to the customer. It begins with the registration of a customer order, followed by the creation of a transport document. The logistics service provider is then contacted to arrange the transport of goods to a seaport, involving booking vehicle capacities and ordering empty containers. The goods are packed into handling units, loaded onto trucks, and transported to a terminal. At the terminal, containers are weighed and stored before being loaded onto vehicles for departure. Occasionally, delays necessitate rescheduling containers for the next available vehicle.

Order Management (OM) [1]. Focuses on the order management processes. This process encompasses the entire cycle of managing customer orders, from registration to shipment. Customers place orders for products, which are then assigned to sales representatives for processing. The sales department confirms orders and processes payments, while the warehousing department checks stock availability and reorders items if necessary. Items are picked, packed, and prepared for shipment. Shipments are managed by a separate team, and the process ensures that deliveries are completed successfully.

Procure-to-Pay (P2P) [3] Details the entire procurement process from purchase to payment. The P2P process starts with the creation of a purchase requisition and proceeds through approval, vendor selection, and purchase order creation. Goods receipts and invoice processing follow, culminating in the payment execution. Special behaviors in the process include maverick buying (unauthorized purchases), duplicate payments, and lengthy approval processes. This process utilizes SAP transactions to simulate real-world scenarios.

Table 3 presents some statistics and specifications of the events logs associated with the case studies. Further information on the synthetic events log can be found at `https://www.ocel-standard.org/event-logs/overview/`.

Following, Table 4, are the object-centric CRs extracted from the presented case studies. CRs, in general, are compliance rules that affect the processes.

2.2 Applicability Demonstration

Following are the Object-Centric Compliance Rules (OCCRs) defined for each CRs aforementioned. The OCCRs can be used with the tool provided as a by-product of our work which is available at `https://anonymous.4open.science/`

Case study	Event types	No. events	Object types	No. objects
LOG	14	35761	7	14013
OM	11	21008	6	10840
P2P	10	14671	7	9543

Table 3. Overview on object-centric sample event logs.

id	Specification	Case study
cr1	A Customer cannot create more than 5 orders	LOG
cr2	A Transport Document is mandatory i.o.t. starting handling an order	LOG
cr3	The handling of an order should not exceed 2 working days	LOG
cr4	In the last week, a maximum of 10 containers is accepted for delayed departure	LOG
cr5	After order is confirmed, a Payment Notification should be delivered if the payment has not been performed within 2 days	OM
cr6	Only "Sales" employees can access customers data	OM
cr7	Does maverick buying (the order is placed without proper approval, and created in the system only after the invoice is received) happen in the process	P2P
cr8	Duplicate Payments: An error leading to the same invoice being paid multiple times	P2P
cr9	Lengthy Approval Process: Delays in approving purchase requisitions or purchase orders, which might lead to operational inefficiencies	P2P

Table 4. Compliance requirements extracted from the case studies.

`r/8601`. In addition to OCCRs, we also outline the needed formulas, informally presented, to calculate compliance metrics.

$$\begin{aligned} OCCR1 &= (P_A^{CR1}, FP_b, \psi^{CR1}, \delta^{CR1}, P_B^{CR1}), \qquad \text{where:} \\ P_A^{CR1} &= (\text{Register Customer Order}, \mathit{true}, =, 1, \text{registered CO}, \\ &\quad \text{Customer Order}, \mathit{true}) \\ FP_b &= \text{exclusive} \\ \psi^{CR1} &= [\text{TD for CO, CR for TD}] \\ \delta^{CR1} &= (>, 0) \\ P_B^{CR1} &= (\text{Drive to Terminal}, \mathit{true}, =, 1, \text{CR moved}, \text{Container}, \mathit{true}) \end{aligned}$$

In addition to $OCCR1$, we need a function grouping the events in M by "Customer" object, this would return a series of subsets of M, and each subset would reflect the events of a single "Customer" object; secondly, we have to use a function to count the cardinality of such subsets. Whenever a count results in more than 5, the respective customer violated the compliance rule on the maximum of orders created.

$$
\begin{aligned}
OCCR2 &= (P_A^{CR2}, FP_b, \psi^{CR2}, \delta^{CR2}, P_B^{CR2}), \quad \text{where:}\\
P_A^{CR2} &= (\textsc{Register Customer Order},\ \mathit{true},\ =,\ 1,\ \textsc{registered CO},\\
&\quad \textsc{Customer Order},\ \mathit{true})\\
FP_b &= \text{xLeadsTo}\\
\psi^{CR2} &= [\text{TD for CO}]\\
\delta^{CR2} &= (>, 0)\\
P_B^{CR2} &= (\textsc{Create Transport Document},\ \mathit{true},\ =,\ 1,\ \textsc{created TD},\\
&\quad \textsc{Transport Document},\ \mathit{true})
\end{aligned}
$$

In $OCCR2$, any couple of events in NM is considered a violation, since it means that the handling of an order has started without a proper transport document created.

$$
\begin{aligned}
OCCR3 &= (P_A^{CR3}, FP_b, \psi^{CR3}, \delta^{CR3}, P_B^{CR3}), \quad \text{where:}\\
P_A^{CR3} &= (\textsc{Book vehicle},\ \mathit{true},\ =,\ 1,\ \textsc{VHS booked for TD},\\
&\quad \textsc{Transport Document},\ \mathit{true})\\
FP_b &= \text{corequisite}\\
\psi^{CR3} &= [\text{CR for TD, TR loads CR}]\\
\delta^{CR3} &= (<, 172800)\\
P_B^{CR3} &= (\textsc{Drive to terminal},\ \mathit{true},\ =,\ 1,\ \textsc{TR moved},\ \textsc{Truck},\ \mathit{true})
\end{aligned}
$$

In the case of $OCCR3$, couples of events in NM represent the orders handled exceeding the specified time.

$$
\begin{aligned}
OCCR4 &= (P^{CR4}, FP_u), \quad \text{where:}\\
P^{CR4} &= (\textsc{Reschedule Container},\ \mathsf{today} - \mathit{timestamp} < 7\ \mathit{days},\\
&\quad =,\ 1,\ \textsc{Rescheduled CO},\ \textsc{Container},\ \mathit{true})\\
FP_u &= \text{occurs}
\end{aligned}
$$

For $OCCR4$, we have to use a function to count directly the cardinality of the entire M set that results from the application of the rule. In that case, the requirement is violated if the count results in more than 10 events. Notably, the term today, in all the OCCR examples, is substituted by the current date when the rule is applied.

$$
\begin{aligned}
OCCR5 &= (P_A^{CR5}, FP_b, \psi^{CR5}, \delta^{CR5}, P_B^{CR5}), \quad \text{where:}\\
P_A^{CR5} &= (\textsc{confirm order},\ \mathsf{today} - \mathit{timestamp} > 2\ \mathit{days},\\
&\quad =,\ 1,\ \textsc{customer},\ \textsc{customers},\ \mathit{true})\\
FP_b &= \text{xLeadsTo}\\
\psi^{CR5} &= [\text{places}]\\
\delta^{CR5} &= (\leq, 216000)\\
P_B^{CR5} &= (\textsc{payment remainder},\ \mathit{true},\ =,\ 1,\ \textsc{order},\ \textsc{orders},\ \mathit{true})
\end{aligned}
$$

Regarding $OCCR5$, assuming 12 hours for the system to react for missing payment, the M set corresponds to compliant cases. Differently, in NM, we can observe late remainders, which corresponds to the $E_{\psi}^{out\delta}$ set.

$$\begin{aligned} OCCR6 &= (P^{CR6}, FP_u), \qquad\qquad \text{where:} \\ P^{CR6} &= (\textsc{confirm order},\ \mathit{true},\ =,\ 1,\ \textsc{sales person},\ \textsc{employees},\ role \neq \text{“}Sales'') \\ FP_u &= \text{occurs} \end{aligned}$$

In $OCCR6$, any event in the M set represents a violation of customers' data privacy preservation.

$$\begin{aligned} OCCR7 &= (P_A^{CR7}, FP_b, \psi^{CR7}, \delta^{CR7}, P_B^{CR7}), \qquad\qquad \text{where:} \\ P_A^{CR7} &= (\textsc{Create Purchase Order},\ \mathit{true},\ =,\ 1,\ \textsc{quotation}, \\ &\quad\ \textsc{quotation},\ \mathit{true}) \\ FP_b &= \text{xLeadsTo} \\ \psi^{CR7} &= [\text{Purchase Order of Quotation}] \\ \delta^{CR7} &= (>, 0) \\ P_B^{CR7} &= (\textsc{Approve Purchase Order},\ \mathit{true},\ =,\ 1,\ \textsc{purchase order}, \\ &\quad\ \textsc{Purchase Order},\ \mathit{true}) \end{aligned}$$

For what concerns $OCCR7$, any couple of events in NM represents a violation, because the approval is mandatory before performing any other activity.

$$\begin{aligned} OCCR8 &= (P_A^{CR8}, FP_b, \psi^{CR8}, \delta^{CR8}, P_B^{CR8}), \qquad\qquad \text{where:} \\ P_A^{CR8} &= (\textsc{Create Invoice Receipt},\ \mathit{true},\ =,\ 1,\ \textsc{invoice receipt}, \\ &\quad\ \textsc{invoice receipt},\ \mathit{true}) \\ FP_b &= \text{precedes} \\ \psi^{CR8} &= [\text{invoice\_receipt\_pm}] \\ \delta^{CR8} &= (>, 0) \\ P_B^{CR8} &= (\textsc{Execute Payment},\ \mathit{true},\ =,\ 1,\ \textsc{payment},\ \textsc{payment},\ \mathit{true}) \end{aligned}$$

In addition to $OCCR8$ we need a grouping function in the M set on the object ‘ìnvoice receipt”, whenever a subset presents more than one couple of events, it means that a double payment has been executed for the same invoice.

$$\begin{aligned} OCCR9 &= (P_A^{CR9}, FP_b, \psi^{CR9}, \delta^{CR9}, P_B^{CR9}), \qquad\qquad \text{where:} \\ P_A^{CR9} &= (\textsc{Create Purchase Requisition},\ \mathsf{today} - \mathit{timestamp} > 2\ \mathit{days}, \\ &\quad\ =,\ 1,\ \textsc{purchase\_requisition},\ \textsc{purchase\_requisition},\ \mathit{true}) \\ FP_b &= \text{precedes} \\ \psi^{CR9} &= [\text{assigned\_materials of PR}] \\ \delta^{CR9} &= (>, 172800) \\ P_B^{CR9} &= (\textsc{Approve Purchase Requisition},\ \mathit{true},\ \geq,\ 1,\ \textsc{material}, \\ &\quad\ \textsc{material},\ \mathit{true}) \end{aligned}$$

Lastly, for $OCCR9$ the lengthy cases are those observable in the M set.

References

1. Knopp, B., van der Aalst, W.M.P.: Order management object-centric event log (2023). https://doi.org/https://doi.org/10.5281/zenodo.8428112
2. Knopp, B., Graves, N.: Logistics object-centric event log (2023). https://doi.org/https://doi.org/10.5281/zenodo.8428084
3. Park, G., Tacke, L.: Procure-to-payment (p2p) object-centric event log (2023). https://doi.org/https://doi.org/10.5281/zenodo.8412920