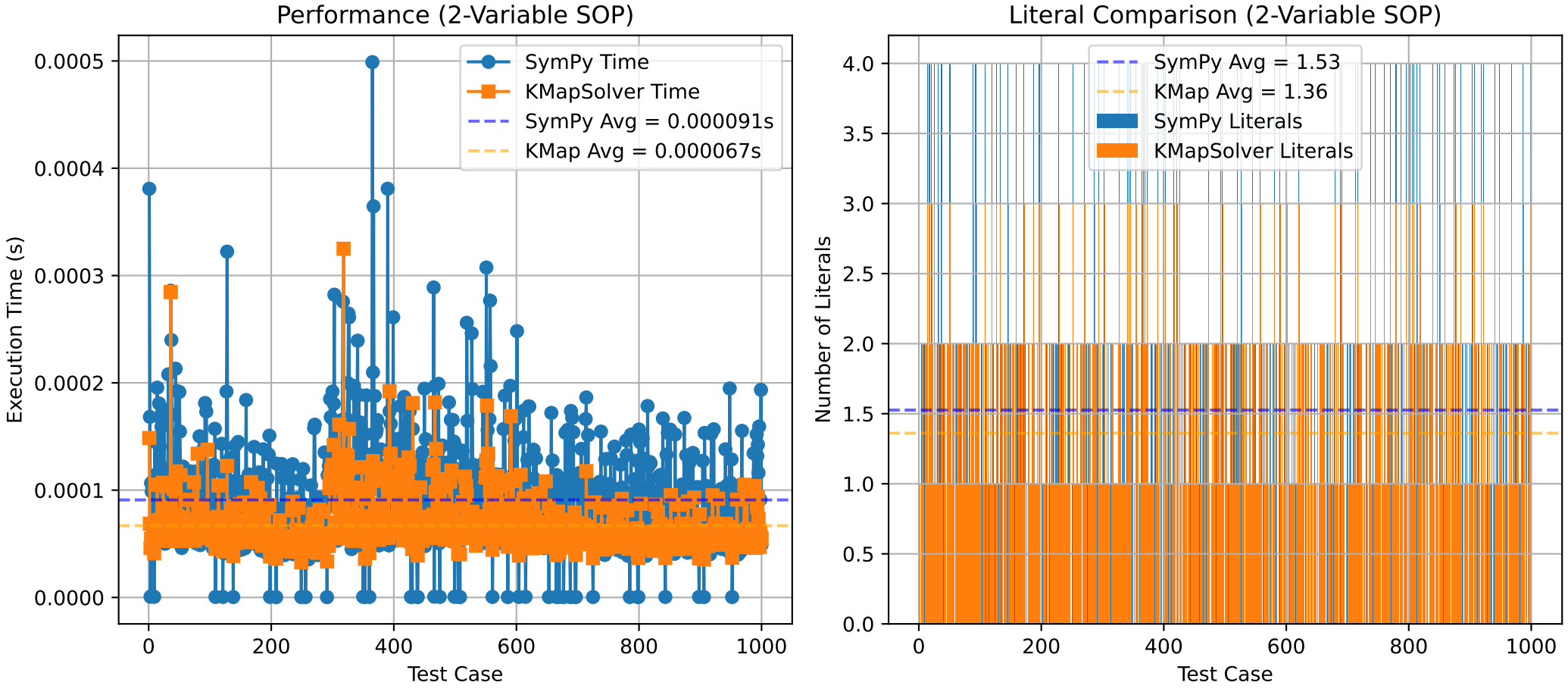
Inference Report

Performance and Simplification Benchmark between SymPy and StanLogic

---

Generated on November 12, 2025


Performance (2-Variable SOP)
SymPy Time
KMapSolver Time
SymPy Avg = 0.000091s
KMap Avg = 0.000067s
Execution Time (s)
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
200
400
600
800
1000
Test Case
Literal Comparison (2-Variable SOP)
SymPy Avg = 1.53
KMap Avg = 1.36
SymPy Literals
KMapSolver Literals
Number of Literals
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
200
400
600
800
1000
Test Case

INFERENCE: 2-Variable SOP

```
================================================================================
```

INFERENCE SUMMARY

```
================================================================================
```

EXECUTION TIME ANALYSIS

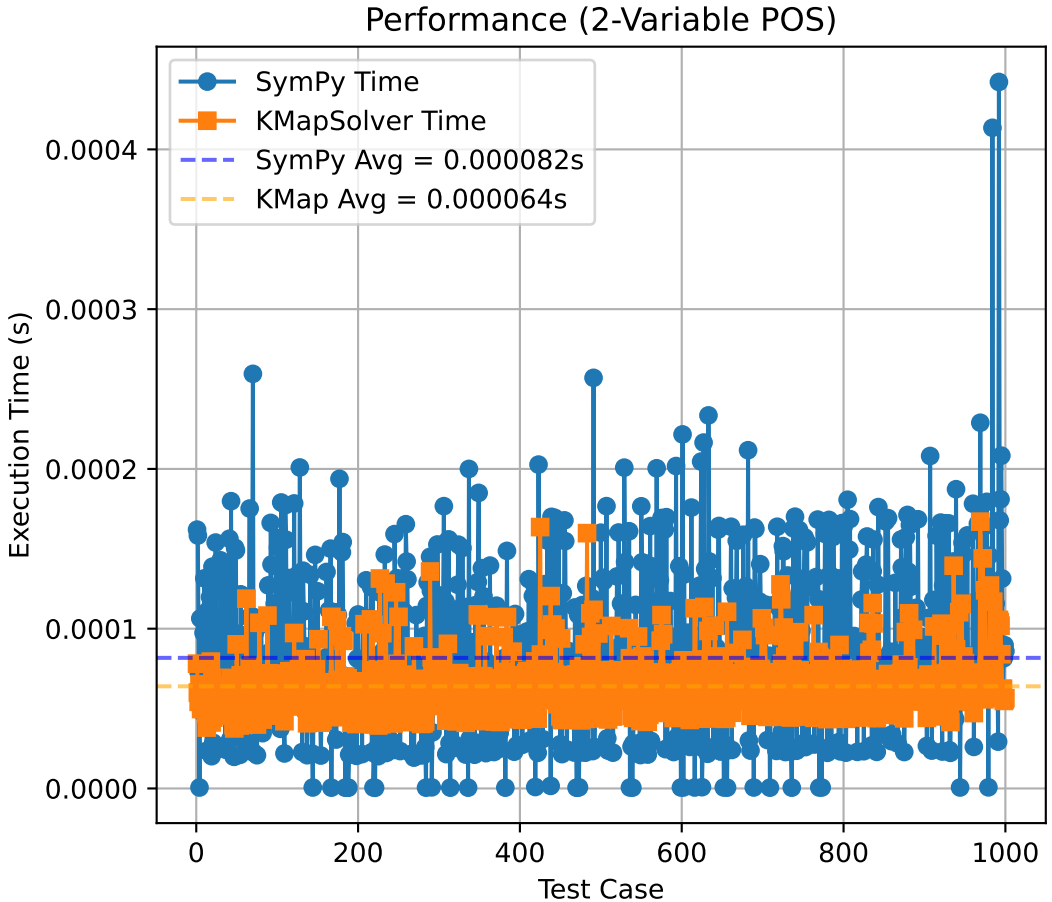
```
  Average SymPy Time:      0.000091 s
  Average KMapSolver Time: 0.000067 s
  Difference:              -0.000024 s (-26.40%)
  Std. Dev (ΔTime):        0.000046 s
  Deviation Ratio:         0.511
  → KMapSolver is faster than SymPy on average.
  → Execution times are stable and consistent.
```

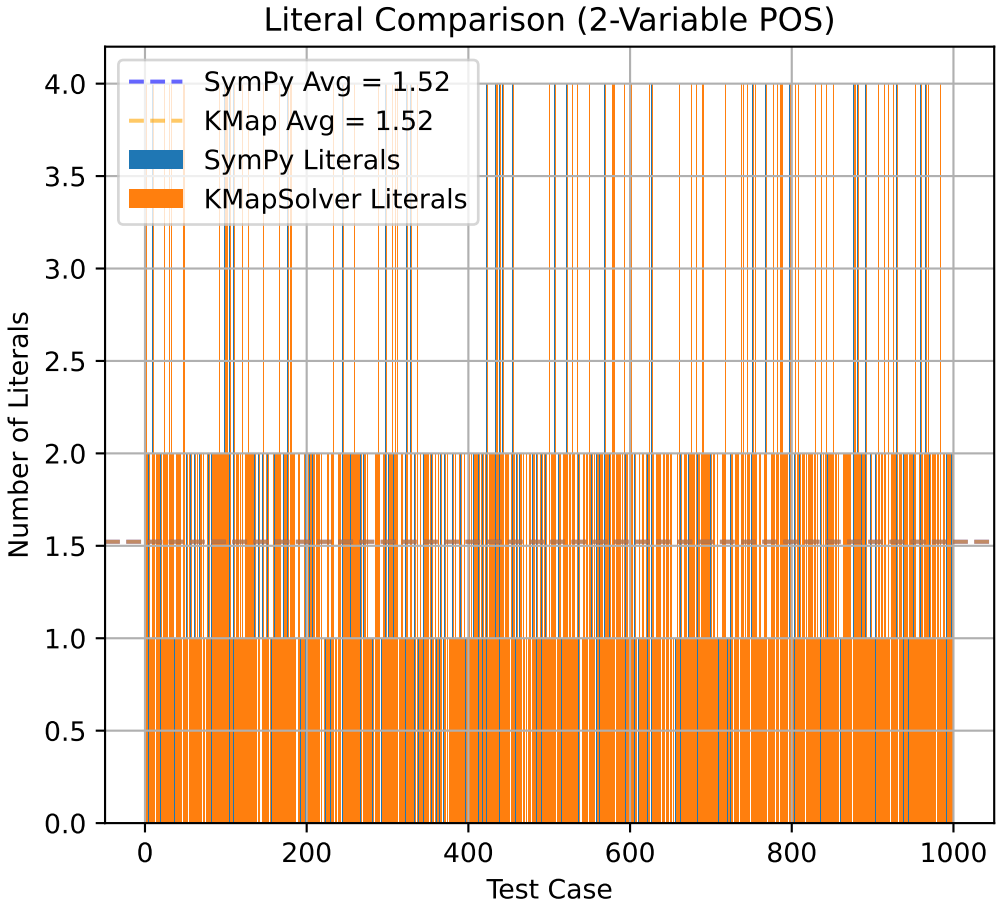
LITERAL COUNT ANALYSIS

```
  Average SymPy Literals:  1.53
  Average KMap Literals:   1.36
  Difference:              -0.17 (-10.8%)
  Std. Dev (ΔLiterals):    0.37
  Deviation Ratio:         0.243
  → KMapSolver produces more minimal logical forms (fewer literals).
  → Literal simplifications are consistent.
```

OVERALL VERDICT

```
  □ KMapSolver achieves comparable or superior simplification efficiency with minimal
time overhead.
================================================================================
```


Performance (2-Variable POS)
SymPy Time
KMapSolver Time
SymPy Avg = 0.000082s
KMap Avg = 0.000064s
Execution Time (s)
0.0000
0.0001
0.0002
0.0003
0.0004
0
200
400
600
800
1000
Test Case


Literal Comparison (2-Variable POS)
SymPy Avg = 1.52
KMap Avg = 1.52
SymPy Literals
KMapSolver Literals
Number of Literals
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0
200
400
600
800
1000
Test Case

INFERENCE: 2-Variable POS

```
================================================================================
```

INFERENCE SUMMARY

```
================================================================================
```

EXECUTION TIME ANALYSIS

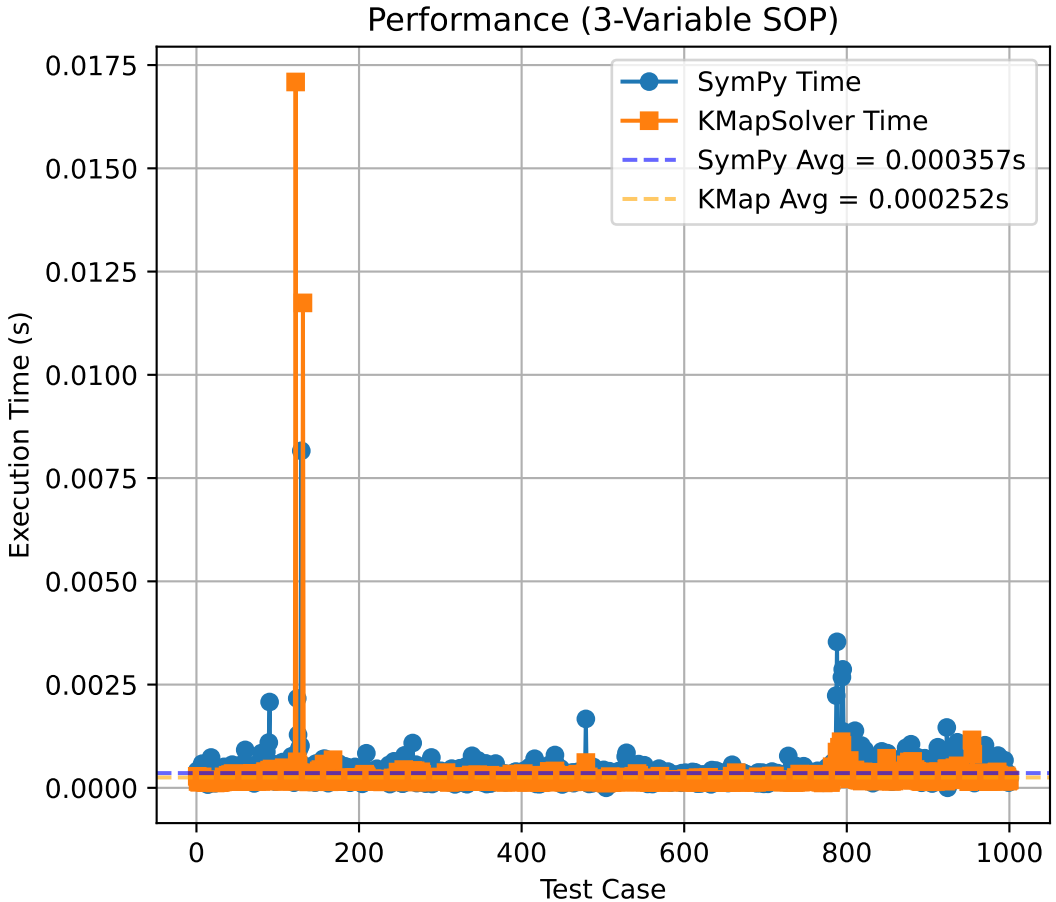
```
  Average SymPy Time:      0.000082 s
  Average KMapSolver Time: 0.000064 s
  Difference:              -0.000018 s (-21.78%)
  Std. Dev (ΔTime):        0.000046 s
  Deviation Ratio:         0.559
  → KMapSolver is faster than SymPy on average.
  → Execution times are stable and consistent.
```

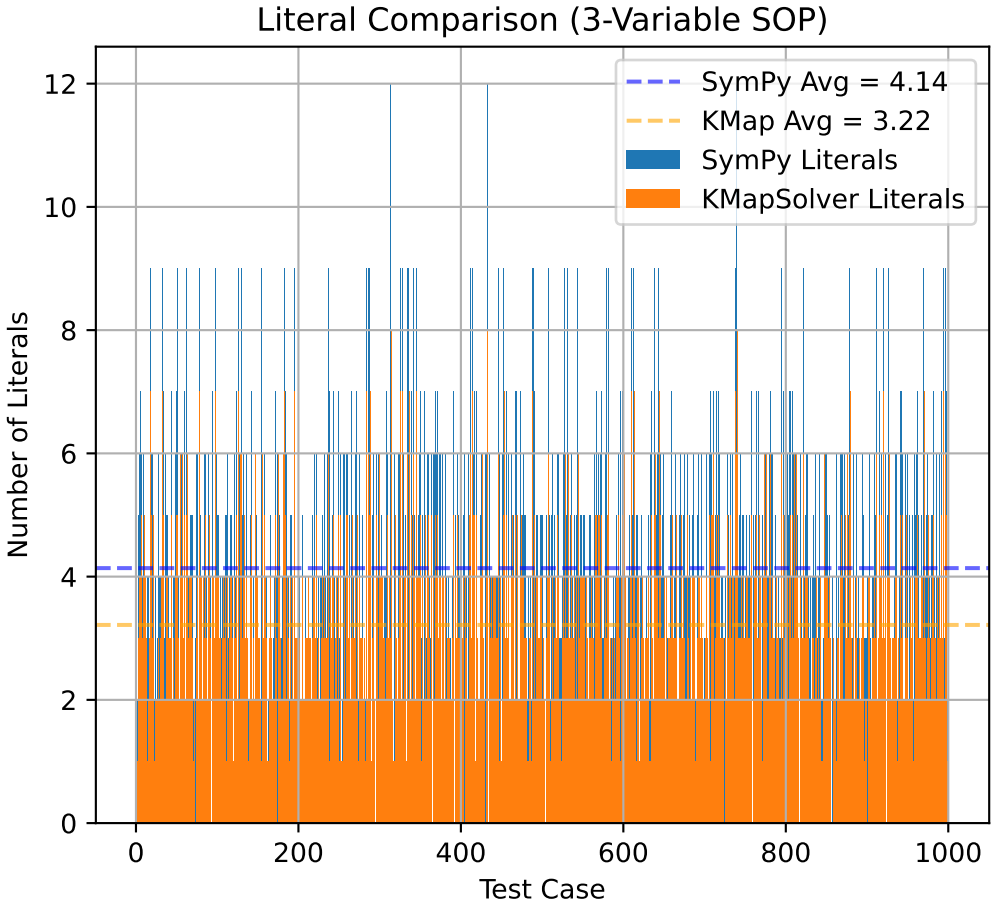
LITERAL COUNT ANALYSIS

```
  Average SymPy Literals:  1.52
  Average KMap Literals:   1.52
  Difference:              +0.00 (+0.0%)
  Std. Dev (ΔLiterals):    0.00
  Deviation Ratio:         0.000
  → Both solvers yield nearly identical simplifications.
  → Literal simplifications are consistent.
```

OVERALL VERDICT

```
  □ KMapSolver achieves comparable or superior simplification efficiency with minimal
time overhead.
================================================================================
```


Performance (3-Variable SOP)
SymPy Time
KMapSolver Time
SymPy Avg = 0.000357s
KMap Avg = 0.000252s
Execution Time (s)
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
200
400
600
800
1000
Test Case


Literal Comparison (3-Variable SOP)
SymPy Avg = 4.14
KMap Avg = 3.22
SymPy Literals
KMapSolver Literals
Number of Literals
12
10
8
6
4
2
0
0
200
400
600
800
1000
Test Case

INFERENCE: 3-Variable SOP

```
================================================================================
```

INFERENCE SUMMARY

```
================================================================================
```

EXECUTION TIME ANALYSIS

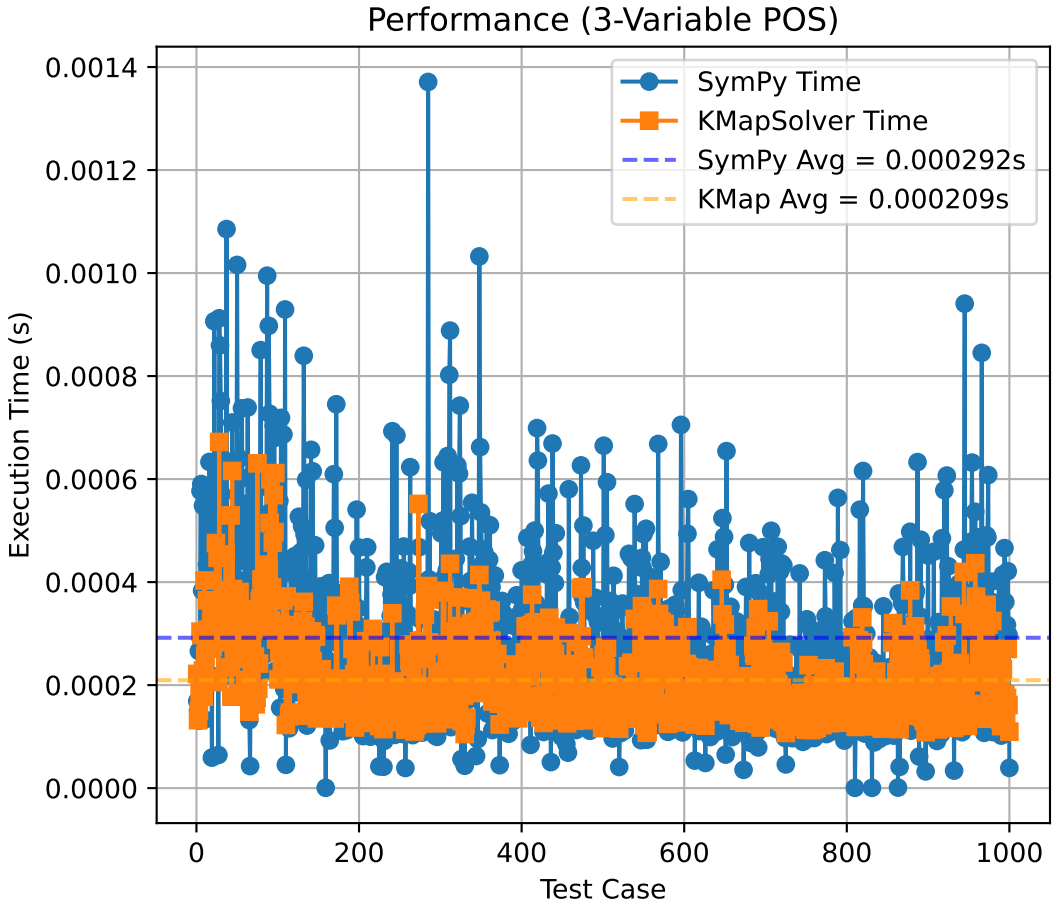
```
  Average SymPy Time:      0.000357 s
  Average KMapSolver Time: 0.000252 s
  Difference:              -0.000105 s (-29.48%)
  Std. Dev (ΔTime):        0.000716 s
  Deviation Ratio:         2.007
  → KMapSolver is faster than SymPy on average.
  → Execution times are stable and consistent.
```

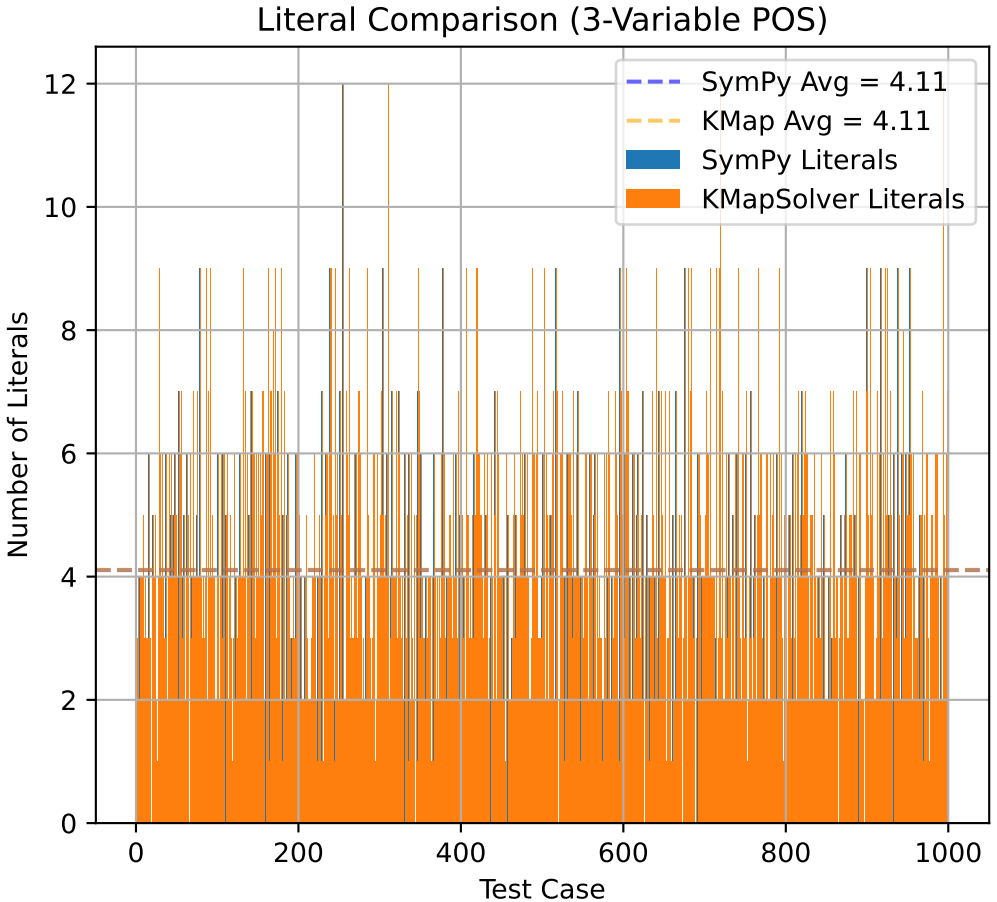
LITERAL COUNT ANALYSIS

```
  Average SymPy Literals:  4.14
  Average KMap Literals:   3.22
  Difference:              -0.92 (-22.3%)
  Std. Dev (ΔLiterals):    0.90
  Deviation Ratio:         0.217
  → KMapSolver produces more minimal logical forms (fewer literals).
  → Literal simplifications are consistent.
```

OVERALL VERDICT

```
  □ KMapSolver achieves comparable or superior simplification efficiency with minimal
time overhead.
================================================================================
```


Performance (3-Variable POS)
SymPy Time
KMapSolver Time
SymPy Avg = 0.000292s
KMap Avg = 0.000209s
Execution Time (s)
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
200
400
600
800
1000
Test Case


Literal Comparison (3-Variable POS)
SymPy Avg = 4.11
KMap Avg = 4.11
SymPy Literals
KMapSolver Literals
Number of Literals
12
10
8
6
4
2
0
0
200
400
600
800
1000
Test Case

INFERENCE: 3-Variable POS

```
================================================================================
```

INFERENCE SUMMARY

```
================================================================================
```

EXECUTION TIME ANALYSIS

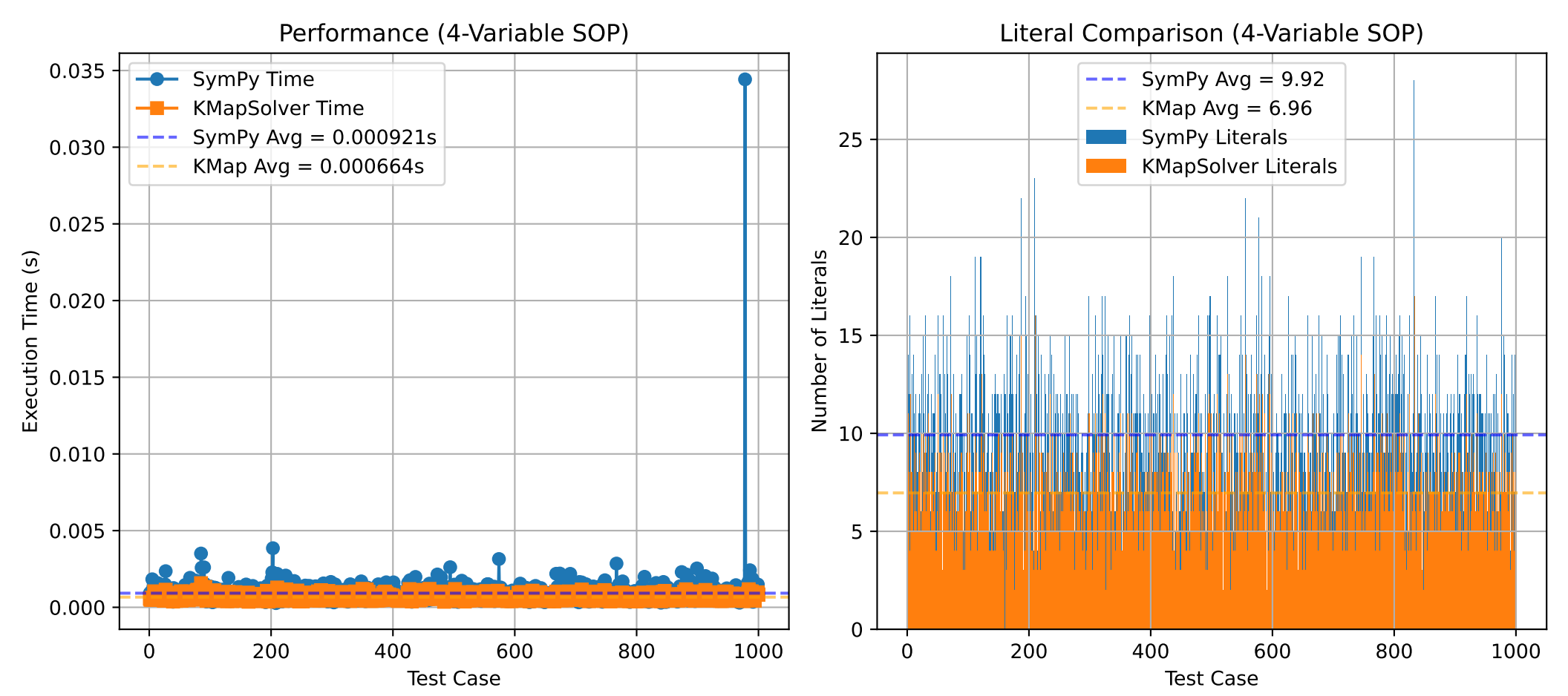
```
  Average SymPy Time:      0.000292 s
  Average KMapSolver Time: 0.000209 s
  Difference:              -0.000082 s (-28.24%)
  Std. Dev (ΔTime):        0.000134 s
  Deviation Ratio:         0.460
  → KMapSolver is faster than SymPy on average.
  → Execution times are stable and consistent.
```

LITERAL COUNT ANALYSIS

```
  Average SymPy Literals:  4.11
  Average KMap Literals:   4.11
  Difference:              +0.00 (+0.1%)
  Std. Dev (ΔLiterals):    0.09
  Deviation Ratio:         0.022
  → Both solvers yield nearly identical simplifications.
  → Literal simplifications are consistent.
```

OVERALL VERDICT

```
  □ KMapSolver outperforms SymPy in runtime while maintaining correctness.
================================================================================
```


Performance (4-Variable SOP)
SymPy Time
KMapSolver Time
SymPy Avg = 0.000921s
KMap Avg = 0.000664s
Execution Time (s)
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
200
400
600
800
1000
Test Case
Literal Comparison (4-Variable SOP)
SymPy Avg = 9.92
KMap Avg = 6.96
SymPy Literals
KMapSolver Literals
Number of Literals
25
20
15
10
5
0
0
200
400
600
800
1000
Test Case

INFERENCE: 4-Variable SOP

```
================================================================================
```

INFERENCE SUMMARY

```
================================================================================
```

EXECUTION TIME ANALYSIS

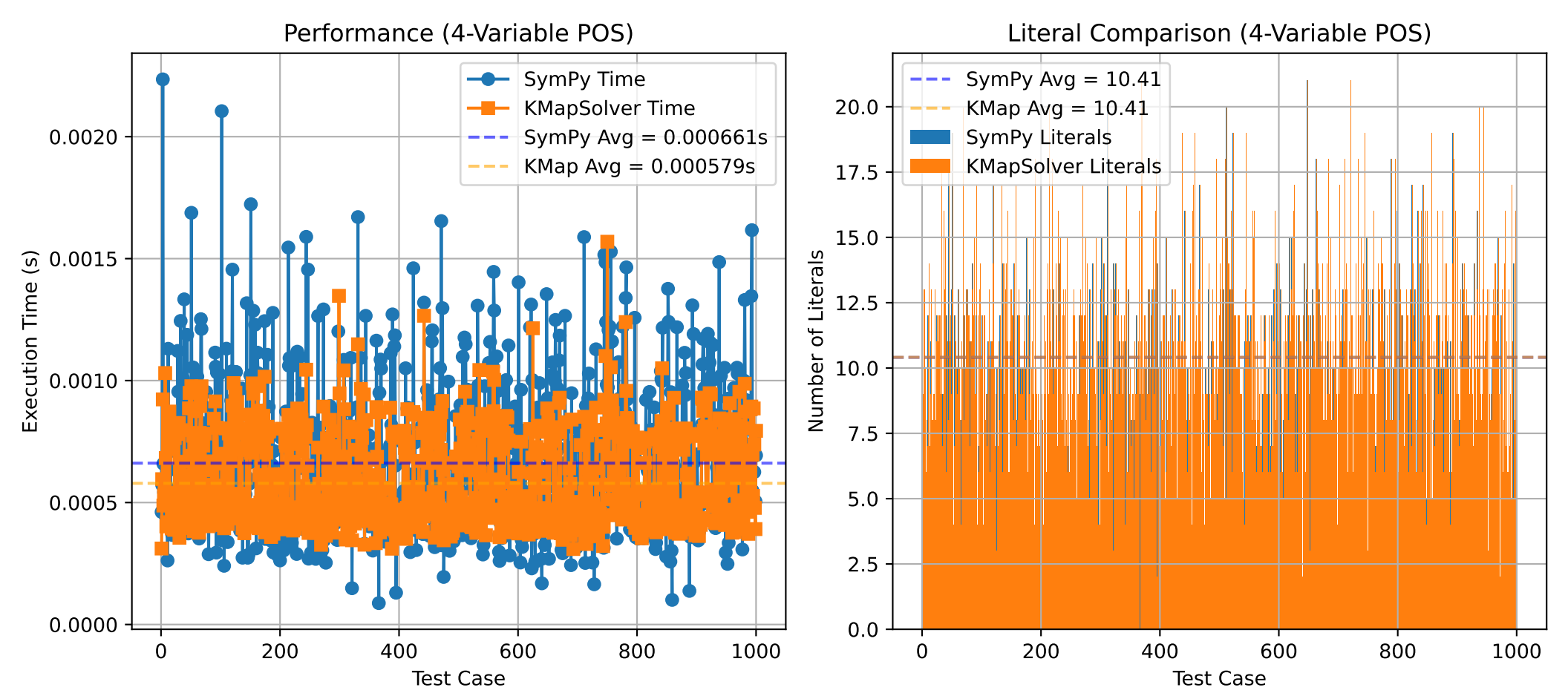
```
  Average SymPy Time:      0.000921 s
  Average KMapSolver Time: 0.000664 s
  Difference:              -0.000258 s (-27.97%)
  Std. Dev (ΔTime):        0.001106 s
  Deviation Ratio:         1.201
  → KMapSolver is faster than SymPy on average.
  → Some variability observed across test runs.
```

LITERAL COUNT ANALYSIS

```
  Average SymPy Literals:  9.92
  Average KMap Literals:   6.96
  Difference:              -2.96 (-29.8%)
  Std. Dev (ΔLiterals):    1.56
  Deviation Ratio:         0.157
  → KMapSolver produces more minimal logical forms (fewer literals).
  → Literal simplifications are consistent.
```

OVERALL VERDICT

```
  □ KMapSolver achieves comparable or superior simplification efficiency with minimal
time overhead.
================================================================================
```


Performance (4-Variable POS)
SymPy Time
KMapSolver Time
SymPy Avg = 0.000661s
KMap Avg = 0.000579s
Execution Time (s)
0.0000
0.0005
0.0010
0.0015
0.0020
0
200
400
600
800
1000
Test Case
Literal Comparison (4-Variable POS)
SymPy Avg = 10.41
KMap Avg = 10.41
SymPy Literals
KMapSolver Literals
Number of Literals
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0
200
400
600
800
1000
Test Case

INFERENCE: 4-Variable POS

```
================================================================================
```

INFERENCE SUMMARY

```
================================================================================
```

EXECUTION TIME ANALYSIS

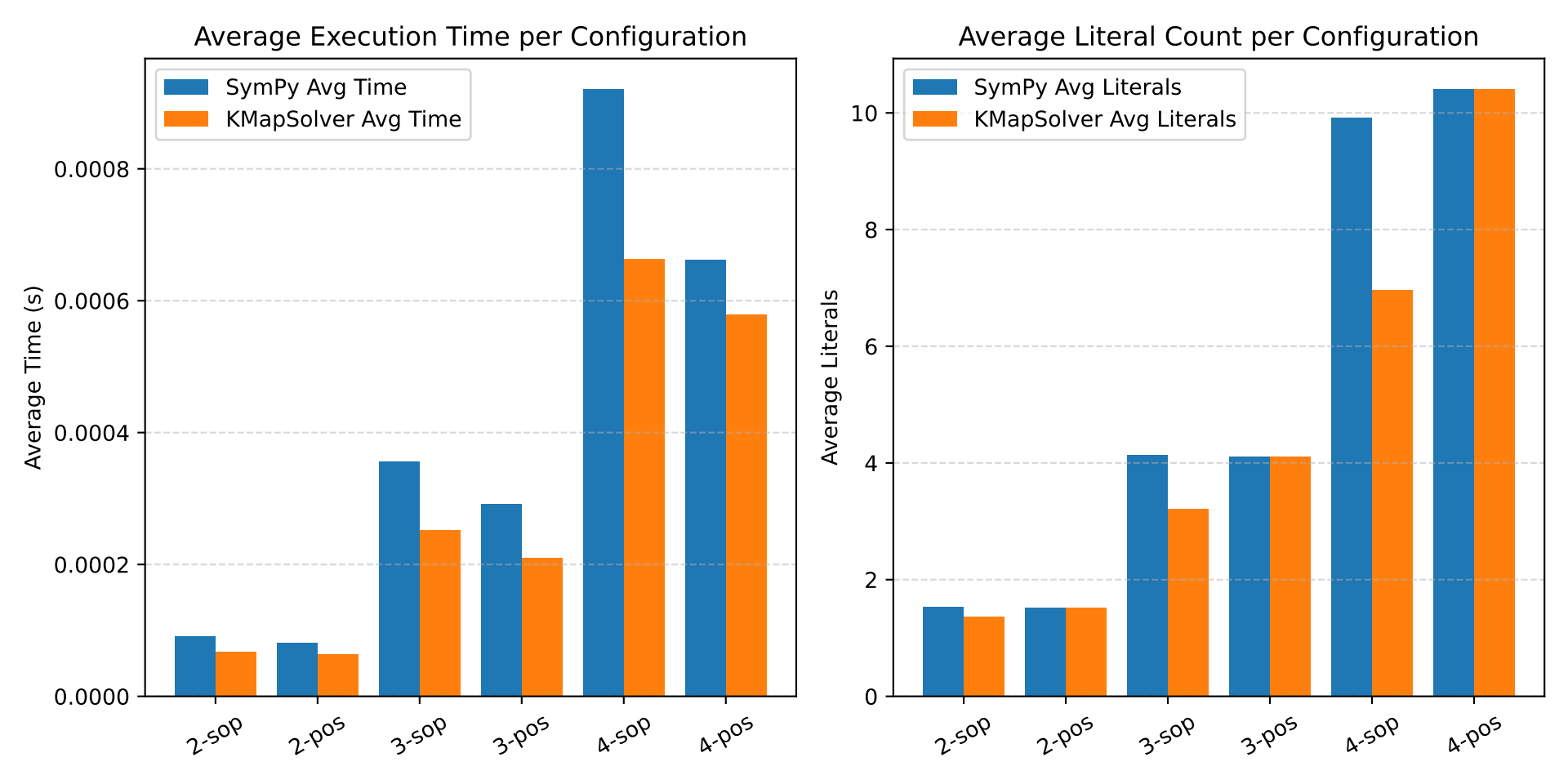
```
  Average SymPy Time:      0.000661 s
  Average KMapSolver Time: 0.000579 s
  Difference:              -0.000083 s (-12.49%)
  Std. Dev (ΔTime):        0.000220 s
  Deviation Ratio:         0.332
  → KMapSolver is faster than SymPy on average.
  → Execution times are stable and consistent.
```

LITERAL COUNT ANALYSIS

```
  Average SymPy Literals:  10.41
  Average KMap Literals:   10.41
  Difference:              +0.00 (+0.0%)
  Std. Dev (ΔLiterals):    0.10
  Deviation Ratio:         0.010
  → Both solvers yield nearly identical simplifications.
  → Literal simplifications are consistent.
```

OVERALL VERDICT

```
  □ KMapSolver outperforms SymPy in runtime while maintaining correctness.
================================================================================
```


Average Execution Time per Configuration
SymPy Avg Time
KMapSolver Avg Time
Average Time (s)
0.0000
0.0002
0.0004
0.0006
0.0008
2-sop
2-pos
3-sop
3-pos
4-sop
4-pos
Average Literal Count per Configuration
SymPy Avg Literals
KMapSolver Avg Literals
Average Literals
0
2
4
6
8
10
2-sop
2-pos
3-sop
3-pos
4-sop
4-pos

OVERALL INFERENCE REPORT

Generated on November 12, 2025

```
================================================================================
```

INFERENCE SUMMARY

```
================================================================================
```

EXECUTION TIME ANALYSIS

```
  Average SymPy Time:      0.000401 s
  Average KMapSolver Time: 0.000306 s
  Difference:              -0.000095 s (-23.70%)
  Std. Dev (ΔTime):        0.000087 s
  Deviation Ratio:         0.217
  → KMapSolver is faster than SymPy on average.
  → Execution times are stable and consistent.
```

LITERAL COUNT ANALYSIS

```
  Average SymPy Literals:  5.27
  Average KMap Literals:   4.60
  Difference:              -0.67 (-12.8%)
  Std. Dev (ΔLiterals):    1.18
  Deviation Ratio:         0.223
  → KMapSolver produces more minimal logical forms (fewer literals).
  → Literal simplifications are consistent.
```

OVERALL VERDICT

```
  □ KMapSolver achieves comparable or superior simplification efficiency with minimal
time overhead.
================================================================================
```