Zeno-Tudor Tomodan, AI @ West University of Timisoara
Faculty of Mathematics and Computer Science
Methods and Practices in Informatics
zeno.tomodan05@e-uvt.ro

Activity #1

A Comparison of SAT Solving Algorithms

Usage Instructions

- **Introduction**

This repository contains implementations of three classical SAT solving algorithms:

- DPLL (Davis-Putnam-Logemann-Loveland)

- Davis-Putnam (DP)

- Resolution-based solver

- **Requirements:**

- Python 3.8 or higher

- `psutil` library (for memory measurement)

- `matplotlib` (for performance plots)

- **Files included:**

- `dpll_solver.py:` DPLL algorithm implementation

- `dp_solver.py:` Davis-Putnam algorithm implementation

- `resolution_solver.py:` Resolution-based solver

- `test.py:` Testing framework for comparing solvers

- **Basic Usage:**

1. To solve a SAT problem using DPLL:

```
from dpll_solver import dpll, generate_random_cnf

cnf = generate_random_cnf(10, 30)    (10 variables, 30 clauses)
result, assignment = dpll(cnf)
print("Satisfiable:", result)
print("Assignment:", assignment)
```

2. To solve using Davis-Putnam:

```
from dp_solver import dp

result = dp(cnf)
print("Satisfiable:", result)
```

3. To solve using Resolution:

```
from resolution_solver import resolution
 result = resolution(cnf)
 print("Satisfiable:", result)
```

- **Testing Framework:**

Run the test script to compare all three solvers:

`test.py:`

This will:

1. Generate random SAT instances
2. Run all three solvers
3. Compare their results
4. Measure and report performance
5. Generate performance plots (plots.png)

- **Performance Evaluation:**

To evaluate DPLL performance with different problem sizes:

```
from dpll_solver import evaluate_performance, plot_performance,
runtimes,memory_usages=evaluate_performance(max_vars=30,step=5,samples=3)
plot_performance(runtimes, memory_usages)
```

- **Note:**

- For large instances (>25 variables), DP and Resolution may be very slow

- DPLL is generally the fastest for practical instances

- All algorithms should agree on satisfiability results