universität freiburg

# Ultimate Automizer

## SV-COMP 2026

**Manuel Bentele** Max Barth Marcel Ebbinghaus Jan Körner Daniel Dietsch
Matthias Heizmann Dominik Klumpp Frank Schüssele Andreas Podelski

April 13, 2026

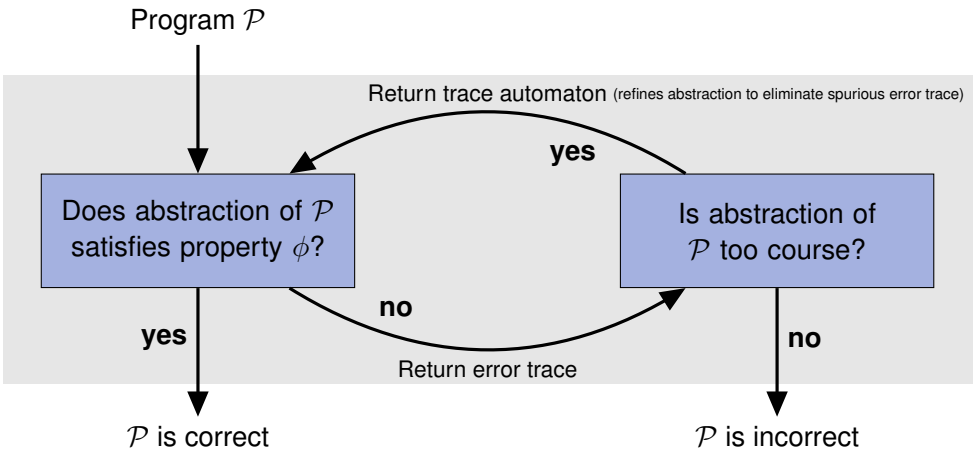
# ULTIMATE


**automata-based software verification**

for **non-reachability**, **memory safety**, **termination**, **overflows**, and **race detection**.

# Verification Approach

## Software Model Checking via Trace Abstraction



# Verification Challenge

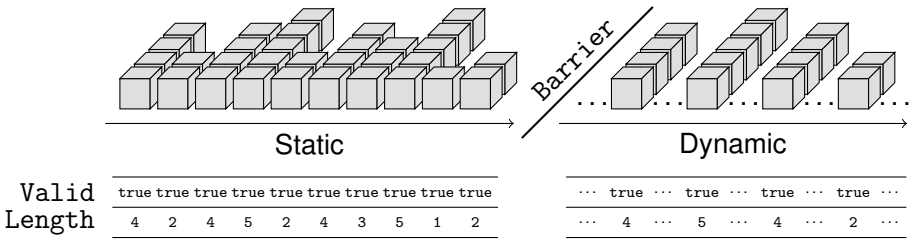
## Represent the Memory of a Program

```
static const uint32_t st_var = 0xED0DDCBA;

void main()
{
  const uint8_t st_arr[4] = {0xBA, 0xDC, 0x0D, 0xED};

  uint8_t *dy_arr = malloc(4 * sizeof(uint8_t));

  memcpy(dy_arr, st_arr, 4 * sizeof(uint8_t));

  for (uint8_t i = 0; i < 4; i++) {
    assert(dy_arr[i] == ((st_var >> (8 * i)) & 0xFF));
    printf("%02x", dy_arr[i]);
  }

  free(dy_arr);
}
```

- Memory blocks
  - Static allocated
  - Dynamic allocated
- Memory operations
  - Management
  - Modification

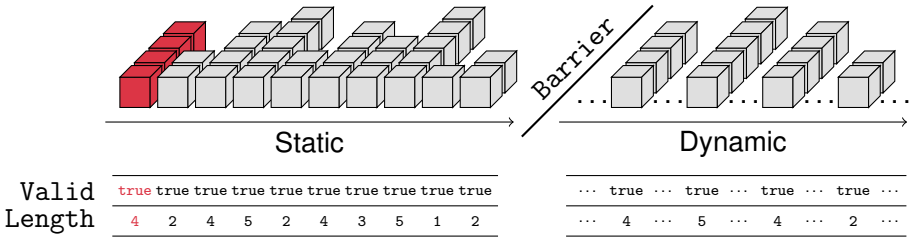
# Memory Modeling

## How ULTIMATE AUTOMIZER Represents Memory (Two-Dimensional Model)



# Memory Modeling

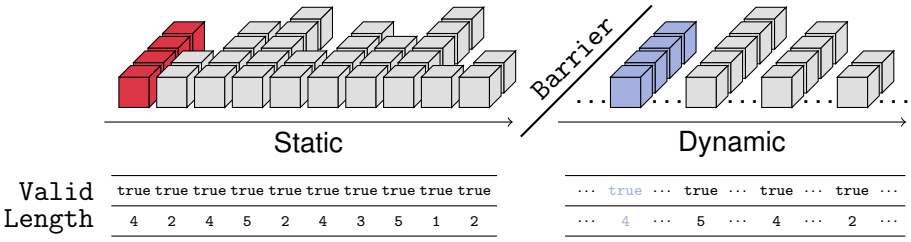
## How ULTIMATE AUTOMIZER Represents Memory (Two-Dimensional Model)




```
const uint8_t st_arr[4] = {0xBA, 0xDC, 0x0D, 0xED};
```

# Memory Modeling

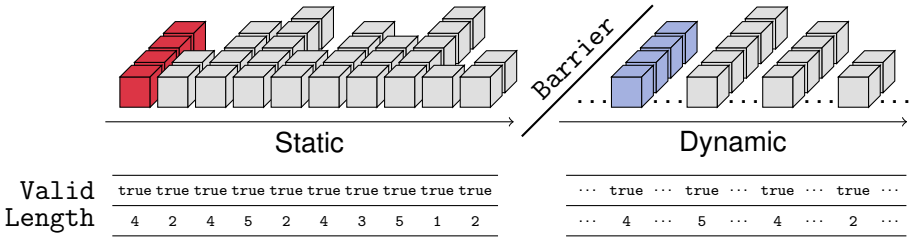
## How ULTIMATE AUTOMIZER Represents Memory (Two-Dimensional Model)




```
uint8_t *dy_arr = malloc(4 * sizeof(uint8_t));
```

# Memory Modeling

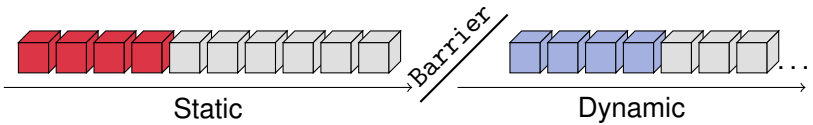
## How ULTIMATE AUTOMIZER Represents Memory (Two-Dimensional Model)




**Such a complex model for reachability analysis?**

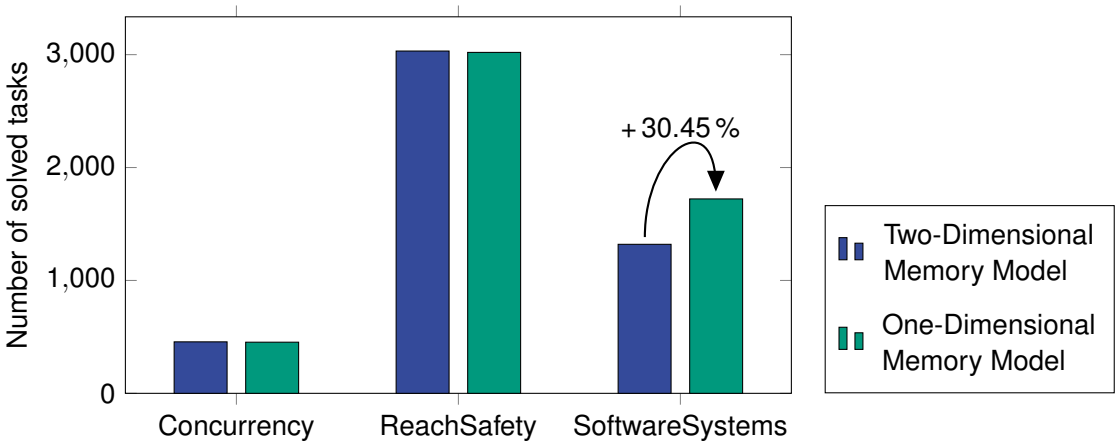
# Recent Feature

## Introduction of a Simplified Memory Model (One-Dimensional Model)



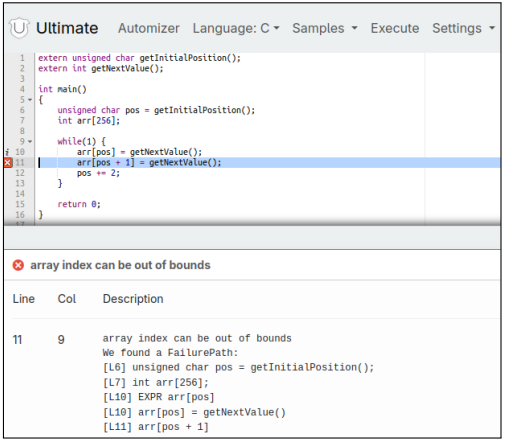
# Performance Evaluation

## Comparison of One-Dimensional Memory Model (SV-COMP 2026 Benchmarks)



# Explore the Tool

## Web Interface of ULTIMATE AUTOMIZER




https://ultimate-pa.org