University of Basel

# Computing Planning Task Width via SAT Compilation

Gadzhiev Khadzhimurad <khad.gadzhiev@stud.unibas.ch>

Department of Mathematics and Computer Science, University of Basel

24th of June 2026

# Introduction

- *width* : measure of the difficulty of a planning task
- Hard to compute
- Can help understand domain properties

# Goal

A theoretical, non-deterministic algorithm was developed to compute the width, we want to :

- Ground this algorithm
- Concrete software implementation
- Automatically compute the width for any planning task

# Meaning

Computing Planning Task Width via SAT Compilation

# Example : Visit-All

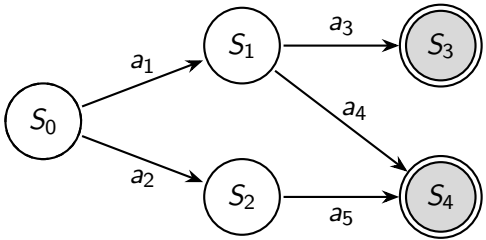
# Example : Visit-All

# Example : Visit-All

# Example : Visit-All

# Generalization : State Space

- States
- Actions
- Transitions
- Initial and Goal states



# Propositional Logic

$$\varphi = (v_1 \wedge \neg v_2) \vee v_3$$

# Propositional Planning Task

- Set of variables $V$
- Operators with effects and preconditions
- States are assignments to variables
- The goal and effects/preconditions are formulas over $V$

# Example : Visit-All

- Cells $A, B$ and $C$
- Variables like : "robot-on-$A$", "visited-$B$", "connected-$A$-$B$"
- Operators like : "move-from-$B$-to-$C$"
- Goal : visit all cells by moving between connected ones.

# Example : Transition

- From { robot-on-$A$, visited-$A$, connected-$c_1$-$c_2$ }
- Via : move-from-$A$-to-$B$
- From { robot-on-$A$, visited-$A$, robot-on-$B$, visited-$B$, connected-$c_1$-$c_2$ }

# Meaning

Computing Planning Task Width via SAT Compilation

# Algorithm : $IW(i)$

Breadth-first search with pruning:

- Compute *novelty* of newly generated states
- Prune states with novelty higher than $i$
- Guaranteed to find optimal plan if $i$ is the width of the task

# (Seemingly) Obvious Solution

Iterate over $i$ until $IW(i)$ finds a solution:

- Start with 1
- Go until $|V|$. Why ?
- Unfortunately, computes **effective** width not **true** width.

# Real Solution

For a planning task $\Pi$, if $IW(i)$ finds a plan, $i$ is not guaranteed to be the width, so we:

- Reject early if $IW(i)$ does **not** find a solution.
- Do additional checking if it does.

# Reachability Graph

Inductively build graph $G^i$ :

- Start with all conjunctions $\leq i$, true in $I$
- Add new nodes only if condition satisfied
- Condition on directed edge between nodes $t_1$ and $t_2$

# Reachability Condition

"All optimal plans for $\Pi(t_1)$ can be extented to be optimal plans for $\Pi(t_2)$ with just one operator."

# Meaning

Computing Planning Task Width via SAT Compilation

# SAT Encoding of Planning Task

"There exists a plan for $\Pi$ of length $T$"

# Example : SAT

"All rooms have a projector."

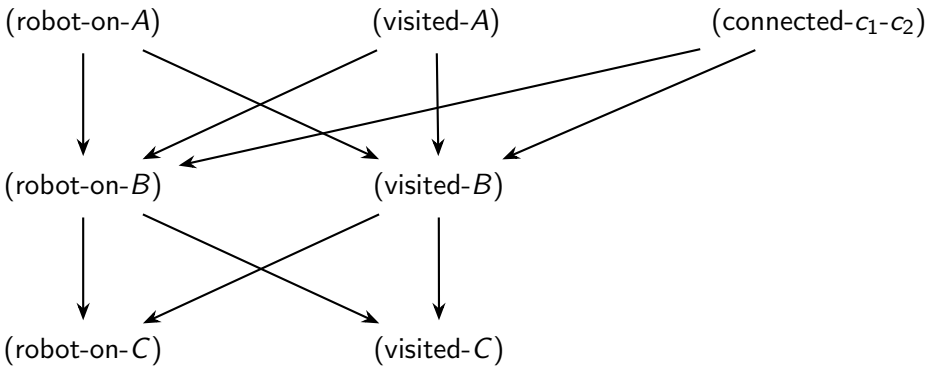
# Example : SAT

"Theres exists a room with no projector."

# Reachability Condition

"There exists an optimal plan for $\Pi(t_1)$ that cannot be extented to be an optimal plan for $\Pi(t_2)$ with just one operator."

# Example : Visit-All




$G^1$ reachability graph

# Example : Visit-All

# Goal Implication

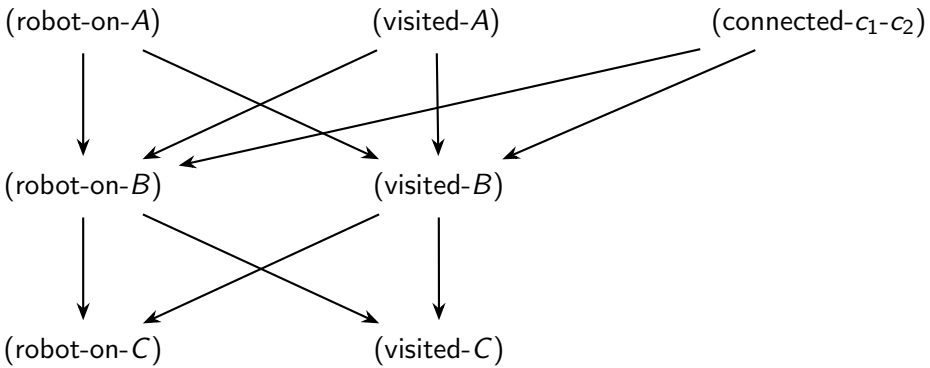
The graph is done, what now ?

- Loop through every node in that graph
- If one of them implies the goal, we found the width

# Goal Implication Condition

"All optimal plans for $\Pi(t)$ are also optimal plans for $\Pi$."

# Example : Visit-All




$G^1$ reachability graph

# Example : Visit-All

# Software used

The implementation was made in Python using the `pyperplan` and `pysat` packages. Problems were in the PDDL format.

# Problems during Implementation

- `pyperplan` SAT encoding couldn't be used
- `pyperplan` parsing code had to be modified
- Built-in `pysat` classes were too memory inefficient
- Encodings are cached

# Experiments

- instances from the `downward-benchmarks` repo
- ran on sciCORE grid with `lab`

# Results

Nir Lipovetzky and Hector Geffner, found upper bounds on the width of certain domains :

- `Blocks World` : 2
- `Logistics` : 2
- `n-puzzle` : 2

# Results

- Low success rate of 6.28%, the rest either timed out or ran out of memory.
- The bounded domains were correct
- Several other domain's effective width was similar

# Results

- `movie` domain had 210 instances
- All succeeded with width of 1
- Conjecture an upper bound on the width

# Future Work

- Algorithmic Optimizations
- Faster language
- Parallelization

Questions?

khad.gadzhiev@stud.unibas.ch

# Backup

| Domain | OOM Error | Timeout | Success |
|---|---|---|---|
| barman-opt14-strips (43) | 36 | 7 | 0 |
| blocks (302) | 18 | 219 | 65 |
| childsnack-opt14-strips (15) | 15 | 0 | 0 |
| depot (188) | 176 | 2 | 10 |
| driverlog (250) | 173 | 44 | 33 |
| grid (19) | 18 | 0 | 1 |
| gripper (460) | 35 | 425 | 0 |
| logistics00 (249) | 151 | 53 | 45 |
| miconic (2325) | 2143 | 167 | 15 |
| movie (210) | 0 | 0 | 210 |
| mystery (45) | 37 | 4 | 4 |
| pipesworld-notankage (259) | 183 | 65 | 11 |
| rovers (558) | 455 | 95 | 8 |
| satellite (1742) | 1735 | 4 | 3 |
| storage (240) | 205 | 23 | 12 |
| tpp (85) | 85 | 0 | 0 |
| visitall-opt11-strips (505) | 284 | 185 | 36 |
| zenotravel (215) | 163 | 21 | 31 |
| **Sum (7710)** | **5912** | **1314** | **484** |
| **Percentage (100%)** | **76.68%** | **17.04%** | **6.28%** |

# Backup

| Domain (Successes) | Minimum width | Average width | Maximum width |
|---|---|---|---|
| barman-opt14-strips (0) | - | - | - |
| blocks (65) | 0 | 0,69 | 1 |
| childsnack-opt14-strips (0) | - | - | - |
| driverlog (33) | 0 | 0,06 | 1 |
| gripper (0) | - | - | - |
| miconic (15) | 2 | 2 | 2 |
| movie (210) | 1 | 1 | 1 |
| pipesworld-notankage (11) | 1 | 1 | 1 |
| rovers (8) | 1 | 1 | 1 |
| storage (12) | 1 | 1 | 1 |
| tpp (0) | - | - | - |
| visitall-opt11-strips (36) | 0 | 0,72 | 1 |
| zenotravel (31) | 0 | 0,06 | 1 |
| **Average (421)** | **0,67** | **0,84** | **1,11** |

# Backup

| **Domain (Successes)** | **Width** |
|---|---|
| depot (10) | 0 |
| grid (1) | 0 |
| logistics00 (45) | 0 |
| mystery (4) | 0 |
| satellite (3) | 0 |

# Backup

- **Miconic** : Out of 650 instances, they found all of them had an effective width of 2. In our case only 15 instances finished computation but all of them reported a width of 2 as well.
- **Visitall** : Out of 21859 instances, they found all of them had an effective width of 1. In our case only 36 instances finished computation but the reported maximum width was indeed 1.
- **Storage** : Out of 240 instances, they found all of them had an effective width of 1. In our case only 12 instances finished computation but all of them reported a width of 1 as well.