# COUNTER-EXAMPLE BASED PLANNING

My algorithm continuously searches counter-examples until finding a plan that no counter-example exists.

**AUTHOR: XIAODI ZHANG**
PhD Candidate
Major in Artificial Intelligence
School of Computing, Australian National University

**SUPERVISOR: ALBAN GRASTIEN**
Senior Lecturer
School of Computing, Australian National University

**SUPERVISOR: HANNA KURNIAWATI**
Associate Professor
School of Computing, Australian National University

**SUPERVISOR: CHARLES GRETTON**
Senior Lecturer
School of Computing, Australian National University

**SUPERVISOR: ENRICO SCALA**
Associate Professor
University of Brescia, Dipartimento di Ingegneria dell'Informazione (DII)

## CONFORMANT PLANNING

### Quiz: How To Reach The Goal In Figure 1?

**Walls**: The robot remains in place when hits the wall.
**Swamp**: The robot cannot leave the swamp (red cell).
**Goal**: The destination robot must reach.
**Robot**: No observation and initially at anywhere except for the swamp cell.

Answer: →*4, ↑*4, →*2, ↓*2, ←*2

No matter where the initial robot's location is, this plan can always lead the robot to arrive at the destination (purple and blue examples in Figure 2).

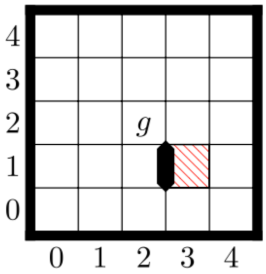


Figure 1

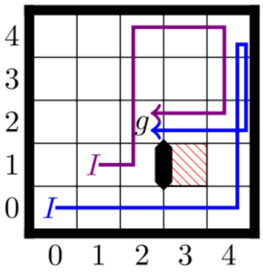


Figure 2

## CPCES

| Iteration 1 | Iteration 2 | Iteration 3 | | Iteration N | |
|---|---|---|---|---|---|
| B = {} | B = {(0,2)} | B = {(0,2), (0,1)} | • • • | B = {(0,2), (0,1), (2,1), ...} | → The Last π is a valid solution |
| π = → | π = ←→→ | π = ←↓↓↑↑→→ | | π = →→→→↑↑↑↑→→↓↓←← | |
| q = (0,2) | q = (0,1) | q = (2,1) | | q = None | |

**B**: Counter-Example Set
**π**: Candidate Plan
**q**: Counter-Example from Candidate Plan

```
Algorithm 1: The conformant planner CPCES.
input: conformant planning problem $P$
output: a conformant plan, or no plan
$B := \emptyset$
loop
    $\pi :=$ produce-candidate-plan$(P, B)$
    if there is no such $\pi$ then
        return no plan
    end if
    $q :=$ generate-counter-example$(P, \pi)$
    if there is no such $q$ then
        return $\pi$
    end if
    $B := B \cup \{q\}$
end loop
```

## FAST DOWNWARD SYSTEM (FD)

### Structure

**Translation**: From PDDL file to SAS+ file
**Searching**: Searching a plan from SAS+ file

### Advantages

Many search engine options
Multi-Valued encoding (SAS+ file)

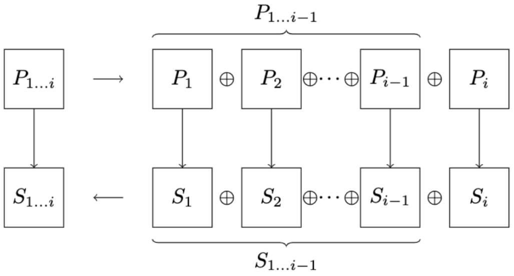
## USING FAST DOWNWARD SYSTEM IN CPCES

### Using FD Directly?

No! Early experiments have shown us that the translation from PDDL to SAS+ file is time consuming and leads to poor SAS+ representations.

### My Idea

Translating each classical planning problem (Pi) separately to SAS+ (Si), and merging them into one SAS+ file.

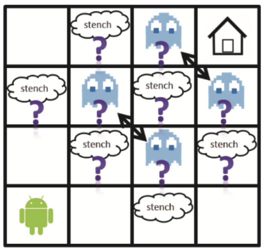


P: PDDL files
S: SAS+ file

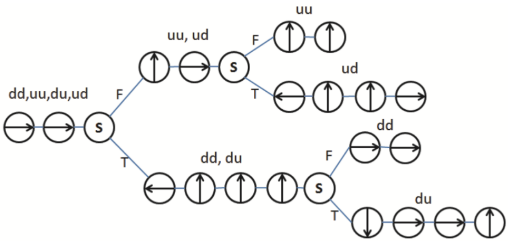
Figure 3

## FUTURE WORKS

### CONTINGENT PLANNING



The robot knows that each Wumpus is hiding in one of two possible locations, but must observe its stench, which carries to all adjacent locations, in order to deduce its whereabouts.



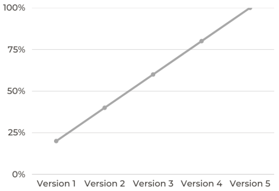
A plan tree, with arrows denoting movement actions, S for sensing a stench, and outgoing edges marked by T or F (stench was observed or not). Branches are associated with a belief — the set of possible states.

Komarnitsky, R., & Shani, G. (2016, March).

### CPCES WEBSITE

Propagating CPCES to the world is my duty, so build a website is necessary. We en- courage everyone to participate in developing CPCES so that CPCES can become one of the most useful planners in the world.

### INCREASING CPCES EFFICIENCY


Version 1
Version 2
Version 3
Version 4
Version 5