Investigating Applicability Heuristics of Answer Set Programming in Game Development

Use Cases and Empirical Study

Evangelos Lamprou Christos Fidas

University of Patras

CHI Greece 2023

Presentation Overview

Problem Statement

Software engineering is hard!

What if, AI code could be placed in modules separate from the rest of the game's logic?

Implementing algorithms is hard!

What if, game logic could be written in a more expressive/higher level language than Python, C++ etc.?

Problem Examples I

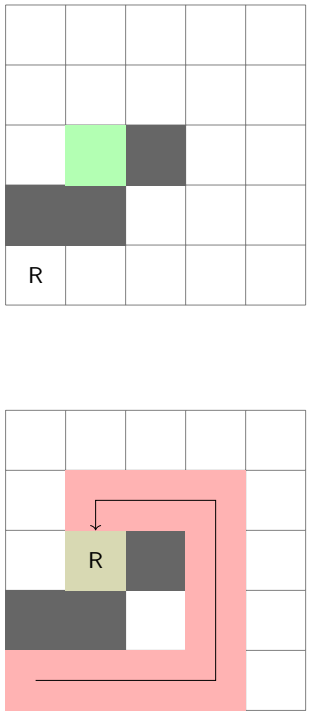
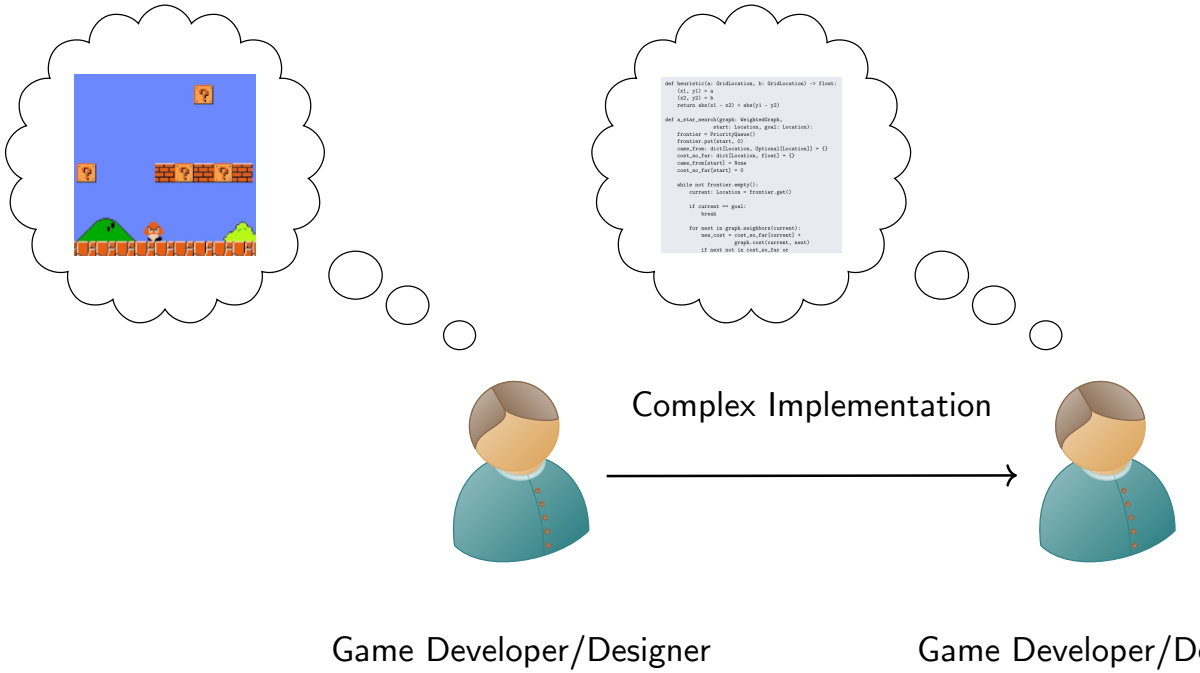


Figure: An optimal path.

The A* algorithm in *Python*

```
def heuristic(a: GridLocation, b: GridLocation) -> float:
    (x1, y1) = a
    (x2, y2) = b
    return abs(x1 - x2) + abs(y1 - y2)

def a_star_search(graph: WeightedGraph,
                  start: Location, goal: Location):
    frontier = PriorityQueue()
    frontier.put(start, 0)
    came_from: dict[Location, Optional[Location]] = {}
    cost_so_far: dict[Location, float] = {}
    came_from[start] = None
    cost_so_far[start] = 0

    while not frontier.empty():
        current: Location = frontier.get()

        if current == goal:
            break

        for next in graph.neighbors(current):
            new_cost = cost_so_far[current] +
                       graph.cost(current, next)
            if next not in cost_so_far or
                new_cost < cost_so_far[next]:
                cost_so_far[next] = new_cost
                priority = new_cost + heuristic(next, goal)
                frontier.put(next, priority)
                came_from[next] = current
    return came_from, cost_so_far
```

Problem Examples II



Answer Set Programming

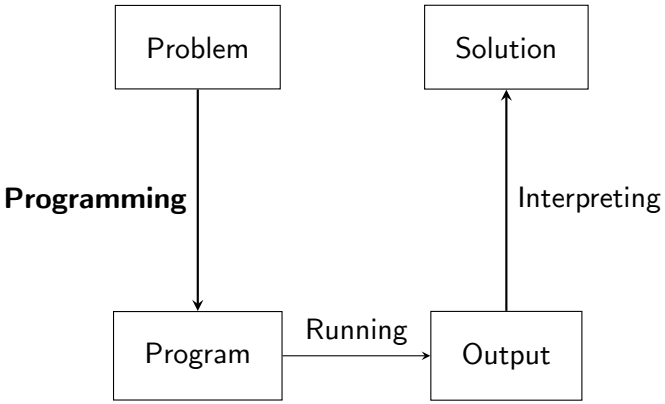


Figure: From problem to solution using **imperative programming**.

Answer Set Programming

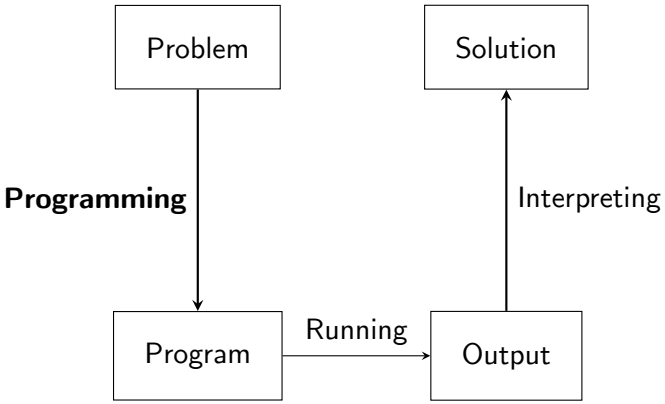


Figure: From problem to solution using **imperative programming**.

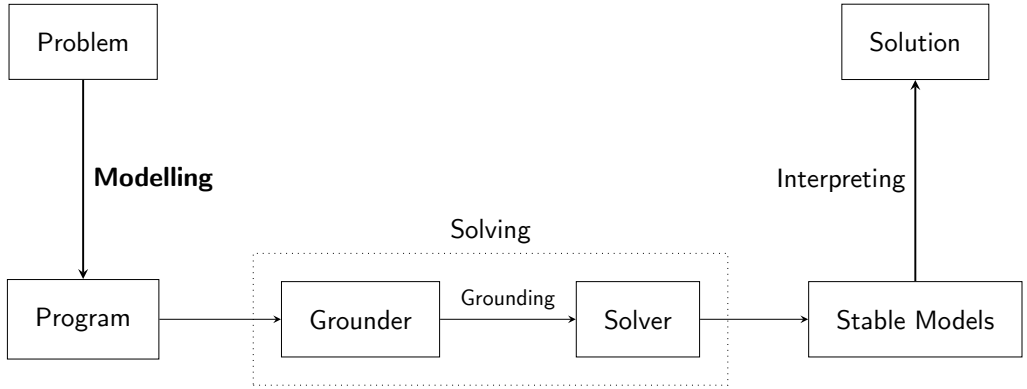
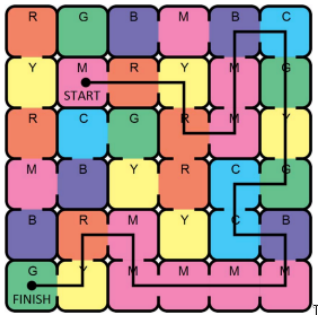


Figure: From problem to solution using **answer set programming**.

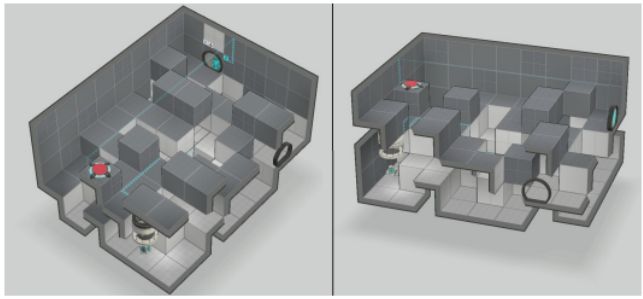
Implementing A* in *Clingo*

```
:- start(S), end(E), not selected(S,_,E) .

:- start(S), selected(_,S,_) .

0{selected(X,E,E)}1 :- edge(X,E,_), end(E) .

0{selected(X,Y,E)}1 :- edge(X,Y,_), selected(Y,_,E) .

cost(E,C) :- C=#sum{W : edge(X,Y,W), selected(X,Y,E)}, end(E) .
#minimize{C : cost(E,C)} .
```

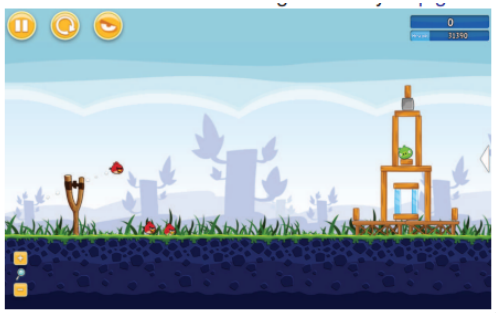
Applications of ASP in games



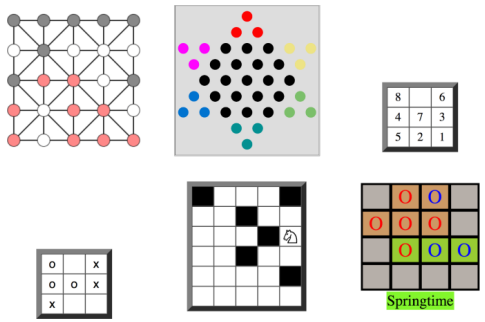
(a) A chromatic maze created with an ASP-based generator [**Smith_2011**].



(b) A level for the game *Portal 2* generated using ASP [**AST_GameLevel_Antonova**].



(c) An ASP-based *Angry Birds* playing agent [**Calimeri_2016**].



(d) General game playing using ASP [**Schiffel_2009**].

Figure: Applications of ASP in Games

Methods

- We propose a framework for applying ASP in games.
- We present applicability heuristics for applying ASP in the implementation of specific game aspects.

Applicability Heuristics

- Brevity
- Relatively Small Solution Space
- Emergent Complexity

Integrating ASP into a Game Development Environment

Game World

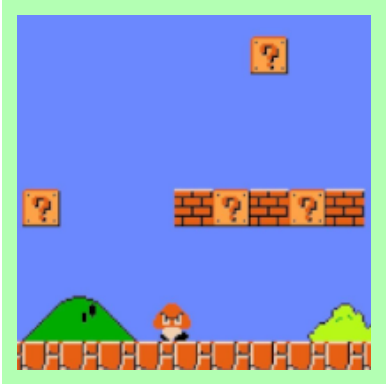


Figure: Integrating ASP into a Game Development Environment

Integrating ASP into a Game Development Environment

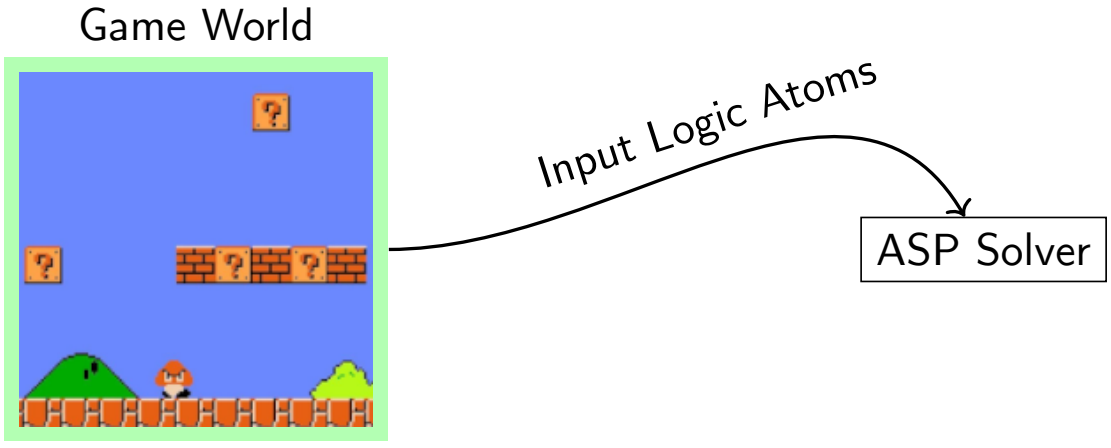


Figure: Integrating ASP into a Game Development Environment

Integrating ASP into a Game Development Environment

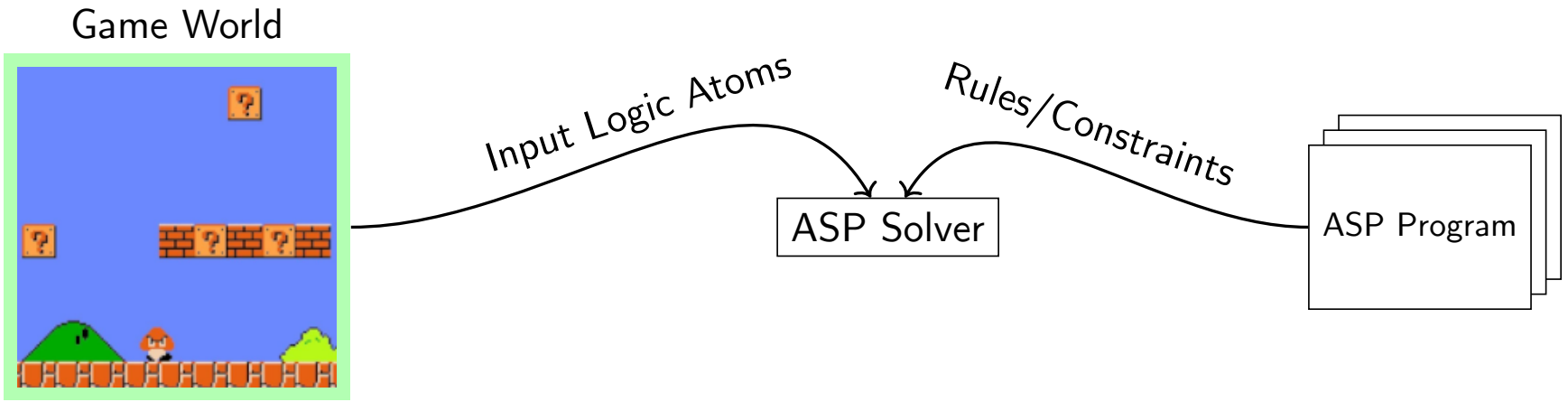


Figure: Integrating ASP into a Game Development Environment

Integrating ASP into a Game Development Environment

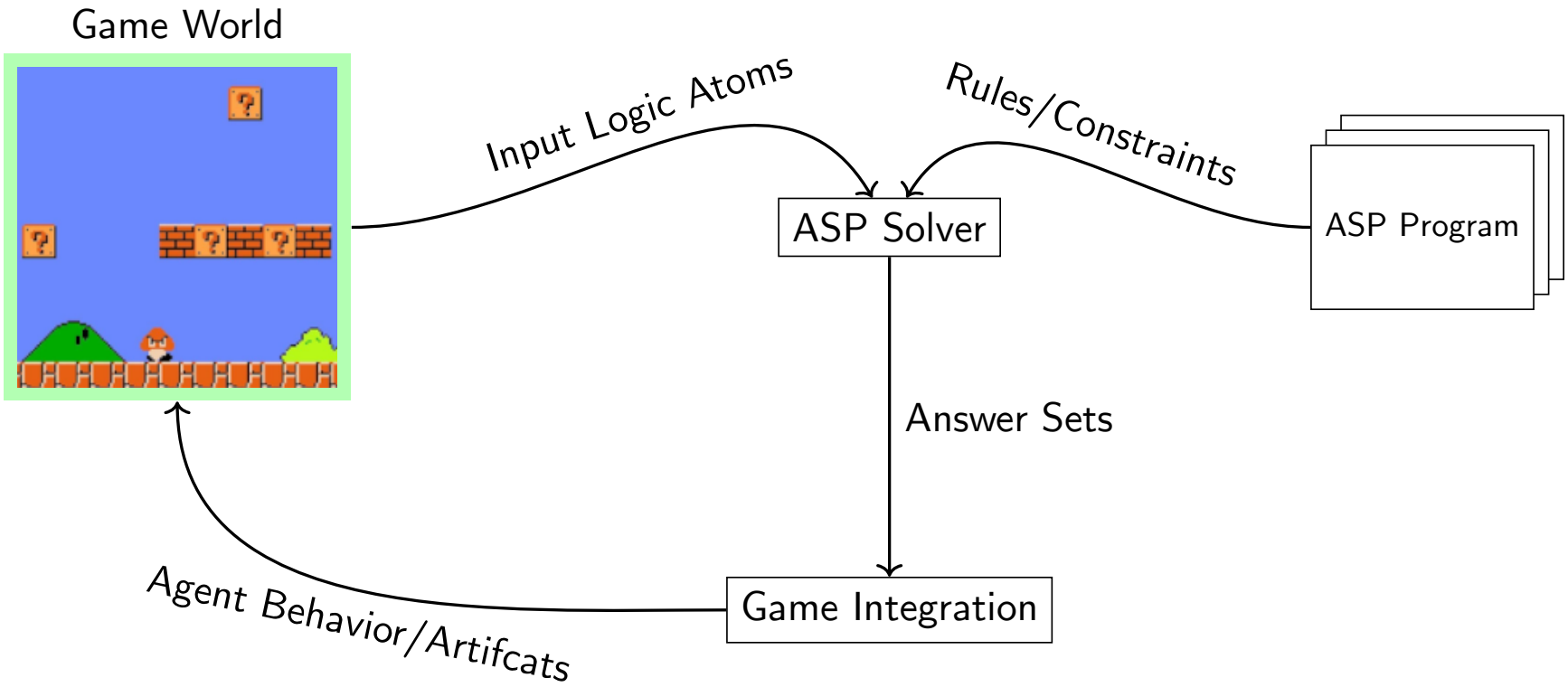


Figure: Integrating ASP into a Game Development Environment

Case Studies

Goals

- Apply the proposed methodology into a wide range of applications.
- Present how the methodology can be applied to each case.

Football (Soccer) Game I

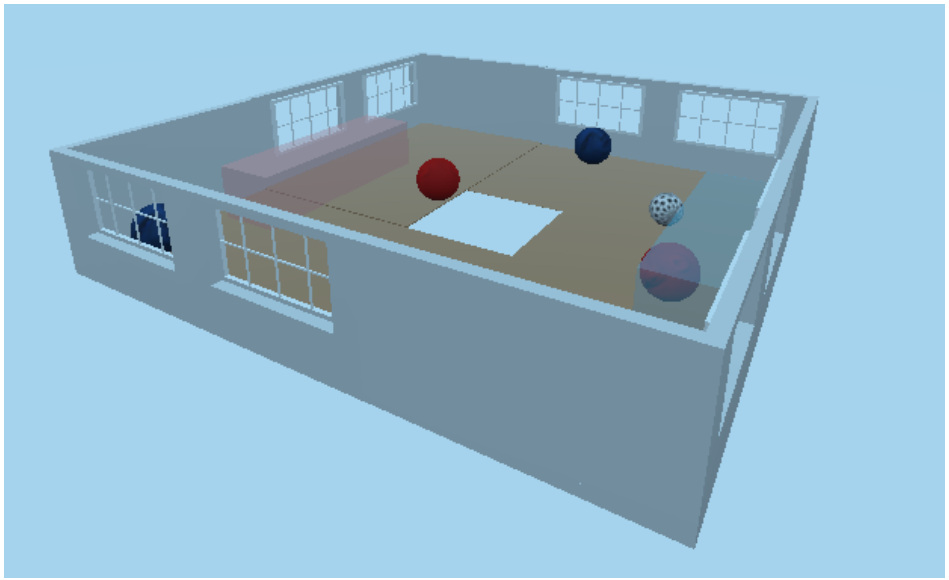


Figure: A top-down screenshot of the football playing field.

Football (Soccer) Game II

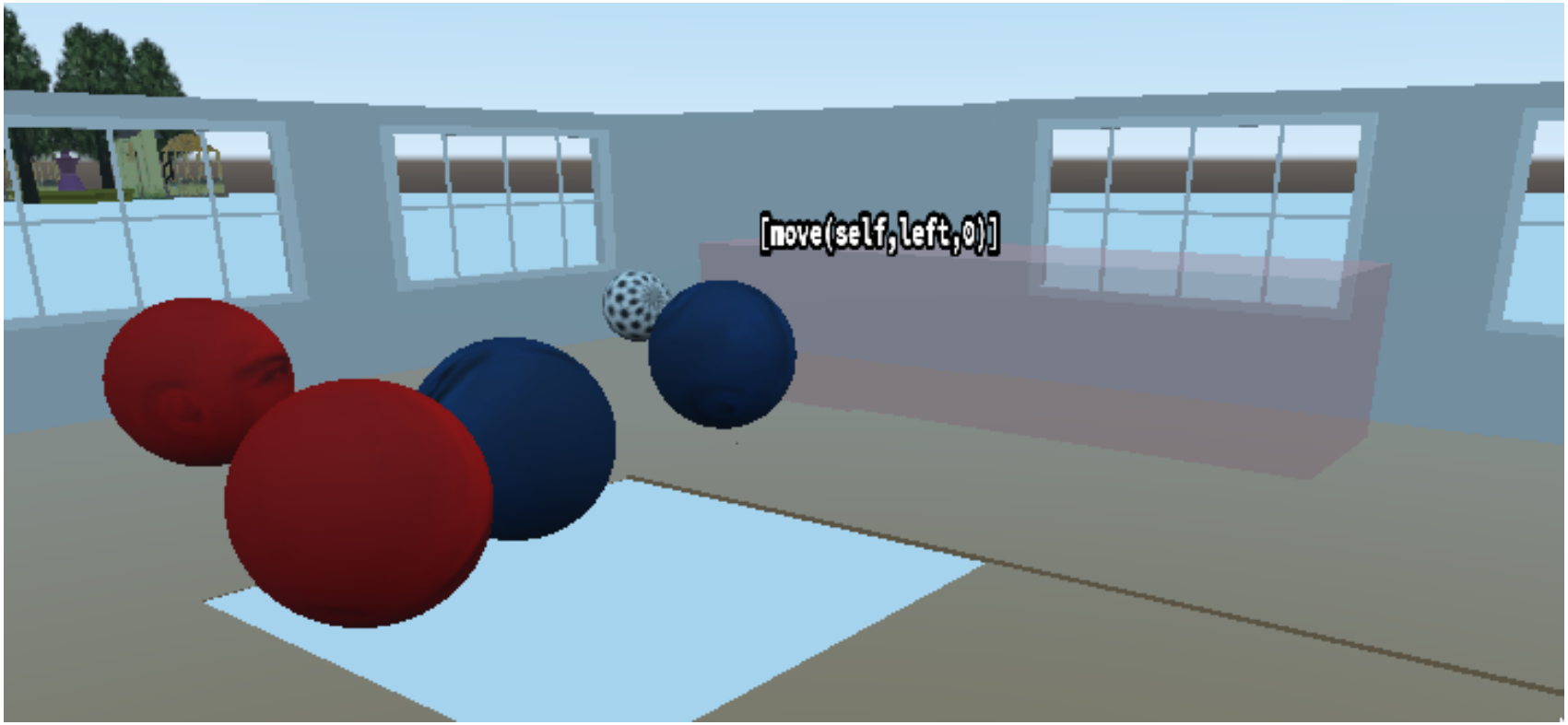


Figure: The football players.

Football (Soccer) Game III

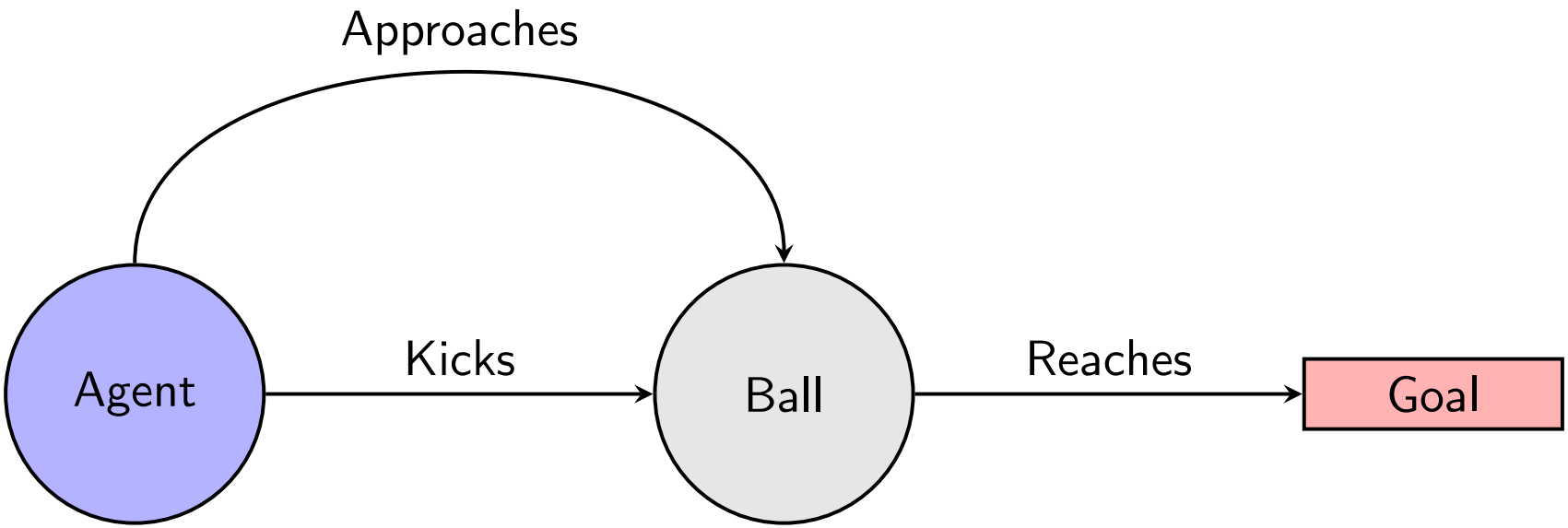
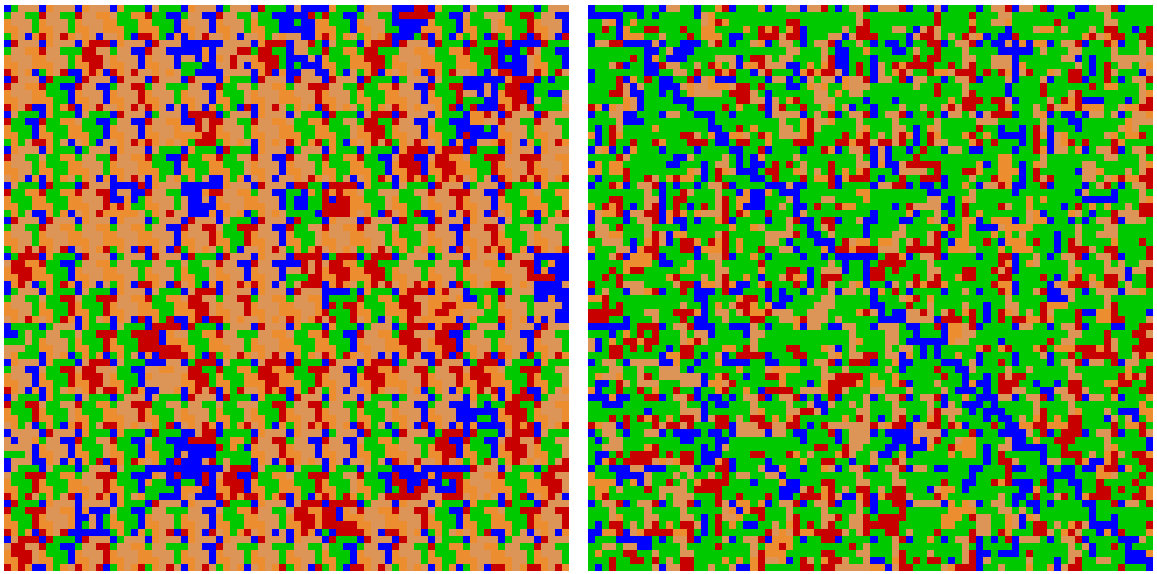


Figure: The agent's reasoning.

Goal

```
#minimize { D : distance(ball, goal, D, t_end + 1) }.
```

Terrain Generation I



(a) A generated map with no constraints encoded.

(b) A generated map with constraints.

Figure: Examples of maps generated using our terrain generator.

User Study

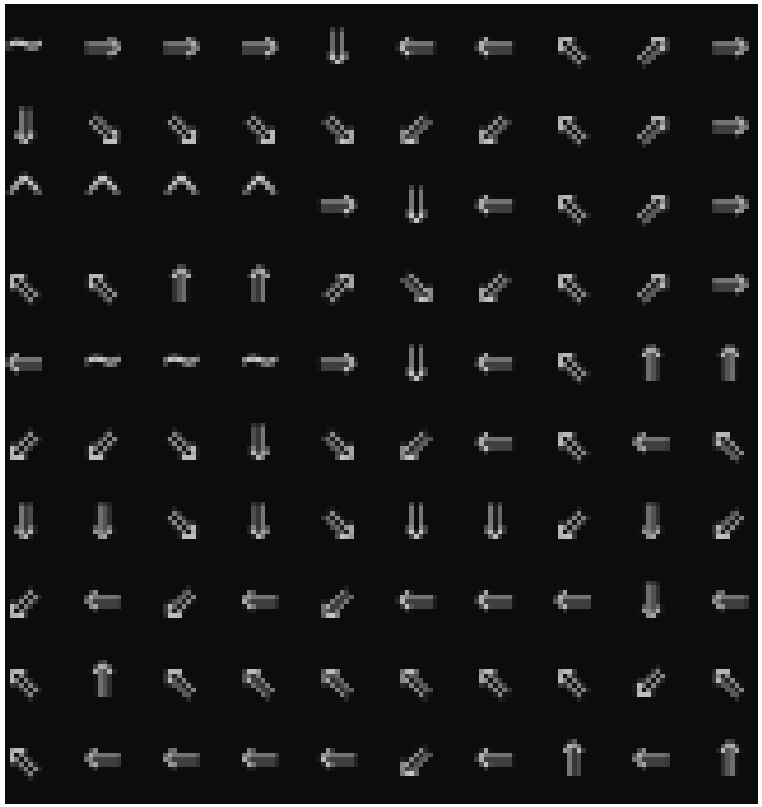
Study Procedure

- **Phase A** - Introduction to ASP
- **Phase B** - Implementation of Game Mechanic/Generation of Content
- **Phase C** - Discussion

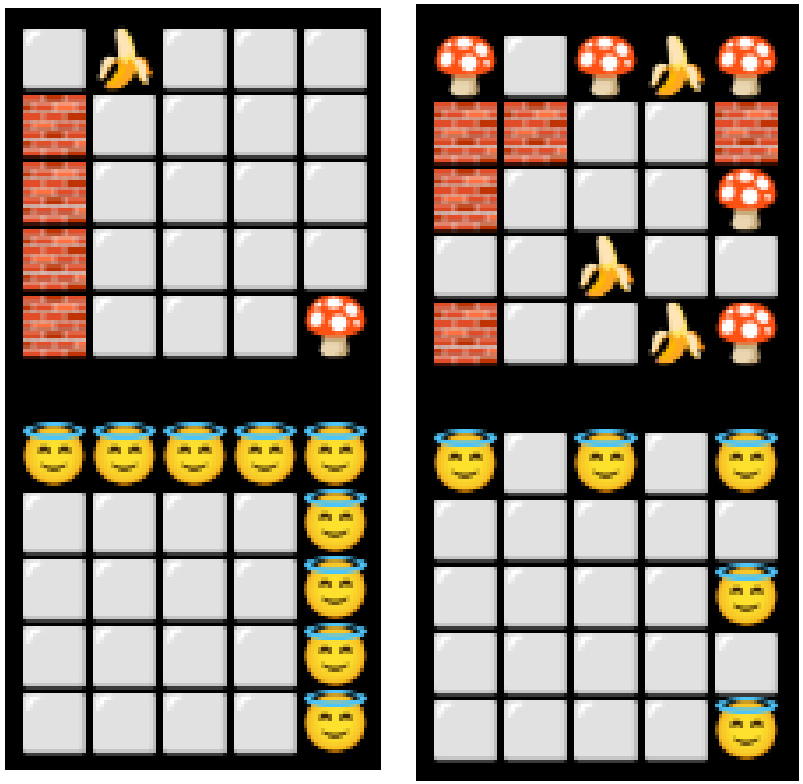
Details

- One to one sessions with the researcher
- 8 participants (6 men, 2 women), ECE Students
- 24 hours in total (0.5 - 3 hours implementation time per participant)

Participant Creations I



(a) Wind direction simulator.



(b) Level generator with difficulty control.

Participant Creations II

(c) Level generator

```
in_chest(diamond,3)
in_chest(gold,1)
in_chest(porkchop,9)

in_chest(diamond,3)
in_chest(gold,3)
in_chest(porkchop,1)

in_chest(diamond,3)
in_chest(gold,3)
in_chest(porkchop,9)
```

(d) Loot generator.

Figure: The applications created by the participants during the study.

Participant Responses I

Does the methodology provide value to the game development process?

- *Participant 1*: 'It gives you the ability to create entirely new game mechanics that you **wouldn't bother developing otherwise**.'
- *Participant 5*: 'Now, it's easier to come up with a game mechanic and think of some constraints for it.'

Would you use ASP again in a future project?

- *Participant 1*: 'The syntax is strange, but I can create a mental model of to use it.'
- *Participant 5*: 'I would use it again in **simple scenarios**.'

Participant Responses II

What could improve the methodology?

- *Participant 7*: 'It would be useful to have some kind of **visualization** that shows how the solver arrives at conclusions.'
- *Participant 8*: 'Maybe an **abstraction layer** built on top of Clingo.'

Conclusions

- ASP can help with
  - Quick prototyping of game mechanics
  - An alternative approach to game development
- Future work
  - Improve usability (visualizations, libraries)
  - Applying the methodology to long-running game development projects

Thank you!

Contact:

e.lamprou@upnet.gr

fidas@upatras.gr

Figure: Code and User Study Data

This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code: T1EDK-T2EDK-01392).