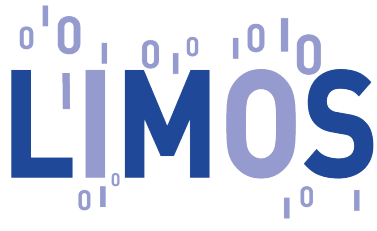
# Reasoning with Ontologies for Non-Player Character Decision-Making in Games

LAMAS & SR 2022, Rennes

**PhD Student**: Sylvain LAPEYRADE
**Supervisors**: Christophe REY, Bruno BACHELET, Loïc YON

LIMOS – Clermont Auvergne University, France

26 august 2022


**Context**: The strategic reasoning of virtual agents is often poor

[1] From reference books in Game AI: Millington [2019], Yannakakis and Togelius [2018] and review article: Simonov et al. [2019]

**Context**: The strategic reasoning of virtual agents is often poor

## Game AI behavioural techniques state of the art[1]

- Ad hoc algorithms (Hard-coded, poor reusability)
- Finite-State Machines (Hard-coded, poor scaling)
- Behavioural Trees (Hard-coded, poor scaling)
- Utility-based AI (Tuning utilities can be laborious)
- Action Planning (Difficult to use, can be expensive)
- Learning-based AI (Resource intensive, black box)

[1] From reference books in Game AI: Millington [2019], Yannakakis and Togelius [2018] and review article: Simonov et al. [2019]

**Context**: The strategic reasoning of virtual agents is often poor

## Game AI behavioural techniques state of the art[1]

- Ad hoc algorithms (Hard-coded, poor reusability)
- Finite-State Machines (Hard-coded, poor scaling)
- Behavioural Trees (Hard-coded, poor scaling)
- Utility-based AI (Tuning utilities can be laborious)
- Action Planning (Difficult to use, can be expensive)
- Learning-based AI (Resource intensive, black box)

**Obversation**: Logic-based methods are almost absent from SOTA

[1] From reference books in Game AI: Millington [2019], Yannakakis and Togelius [2018] and review article: Simonov et al. [2019]

Declarative logic programming helps intuitive rules representation:

## Transformation of a game rule

From our game prototype: "A **human** in the same room as a **monster** or a **pit**, dies"

Listing 1: The same rule in Prolog

```
1 dies(Agent):-
2     human(Agent),
3     human(X, Y),
4     (
5         monster(X, Y) ; pit(X, Y)
6     ).
```

1. $X$ is an element if $X$ is an object or a being
2. $X$ is a being if $X$ is an animal or a monster
3. $X$ is a monster if $X$ is a dragon
4. $X$ is an animal if $X$ is a dog

Listing 2: Simple Ontology in Prolog

```
1 element(X):- object(X). %1
2 element(X):- being(X). %1
3
4 being(X):- animal(X). %2
5 being(X):- monster(X). %2
6
7 monster(X):- dragon(X). %3
8 animal(X) :- dog(X). %4
```

# Stable Model Semantics

For representing the agents knowledge, we can use the *Stable Model Semantics* from Gelfond and Lifschitz [1988] with Negation as Failure (NAF):

# Stable Model Semantics

For representing the agents knowledge, we can use the *Stable Model Semantics* from Gelfond and Lifschitz [1988] with Negation as Failure (NAF):

## Stable Model Semantics

$p$

$r \leftarrow p, q$

$s \leftarrow p, \neg q$

# Stable Model Semantics

For representing the agents knowledge, we can use the *Stable Model Semantics* from Gelfond and Lifschitz [1988] with Negation as Failure (NAF):

## Stable Model Semantics

$p$

$r \leftarrow p, q$

$s \leftarrow p, \neg q$

## Truth Table

| $p$ | $q$ | $r$ | $s$ |
|---|---|---|---|
| True | False | False | True |

# Stable Model Semantics

For representing the agents knowledge, we can use the *Stable Model Semantics* from Gelfond and Lifschitz [1988] with Negation as Failure (NAF):

## Stable Model Semantics

$p$

$r \leftarrow p, q$

$s \leftarrow p, \neg q$

## Truth Table

| $p$ | $q$ | $r$ | $s$ |
|---|---|---|---|
| True | False | False | True |

But how do we represent represent partial or negative knowledge?

# Well-Founded Semantics

- The *Well-Founded Semantics* (WFS) from Van Gelder et al. [1991] is a 3-valued version of the *Stable Model Semantics*
- In the WFS a proposition can be *true*, *false* or *undefined*
- *Undefined* values are used for ignorance or contradictory info

Listing 3: Use of the Well-Founded Semantics in SWI-Prolog

```
1 is_true(Atom):- call_delays(Atom, true).
2 is_false(Atom):- \+ Atom.
3 is_undefined(Atom):-
4     call_delays(Atom, Condition),
5     Condition \== true.
```

# WFS with Partial Knowledge

## With partial knowledge of the truth values of atoms

- Propositions with *true* atoms are *true*
- Propositions with *false* atoms are *false*
- If $p$ is *true* then $\neg p$ is *false*
- Conversely, if $p$ is *false* then $\neg p$ is *true*

## Assume *undefined* propositions *false* and see what holds

- Any proposition symbols not in the resulting set must be *false*
- We can then make them false in the well-founded model
- This operation can be repeated until no value changes

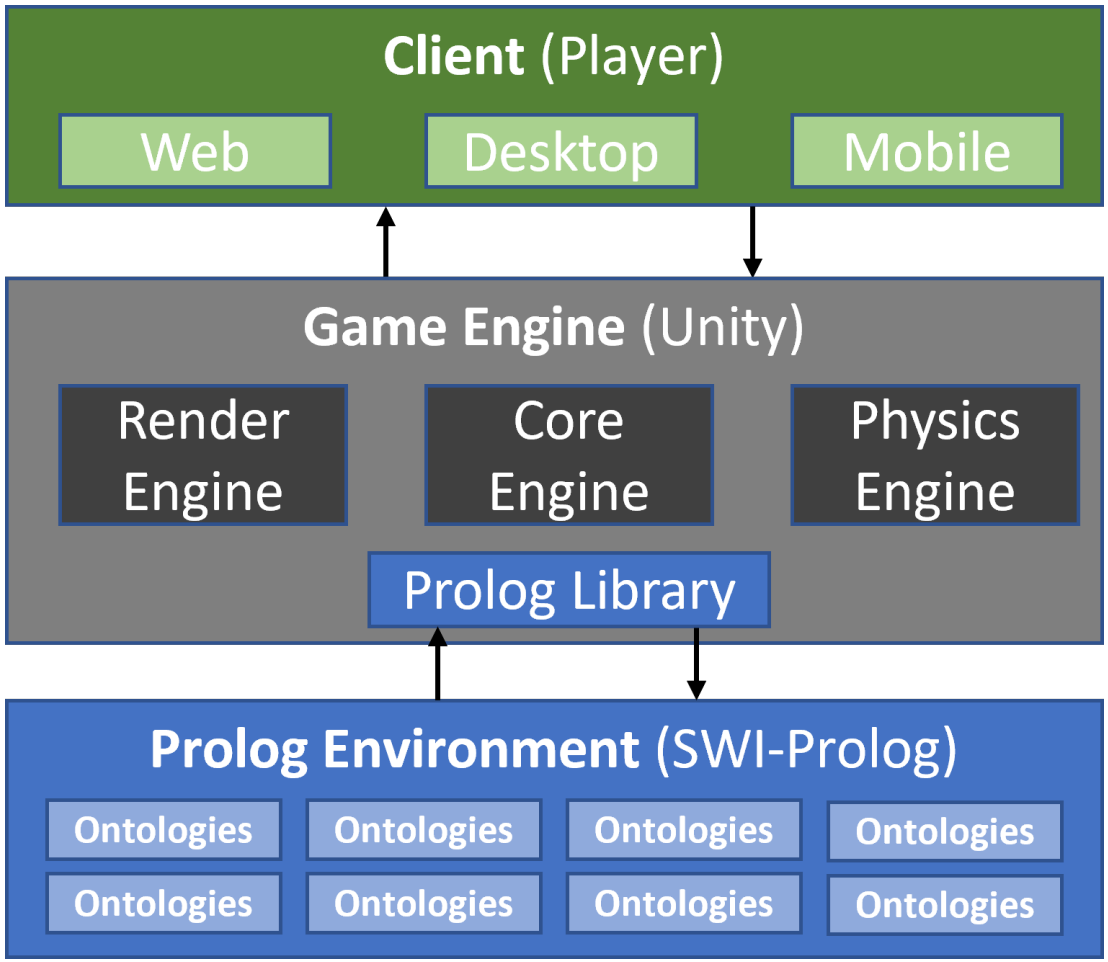
# WFS Example

Listing 4: Representing False Facts in SWI-Prolog

```
1  monster(X,Y):-
2      pos_monster(X,Y),
3      tnot(neg_monster(X,Y)).
4
5  monster(X,Y):- % Positive Contradiction
6      pos_monster(X,Y),
7      neg_monster(X,Y),
8      tnot(monster(X,Y)).
9
10 monster(X,Y):- % Negative Contradiction
11     tnot(pos_monster(X,Y)),
12     tnot(neg_monster(X,Y)),
13     tnot(monster(X,Y)).
```

*Tnot* denotes *Tabled Negation*, it checks if there is an atom whose truth cannot be proven without making a call to a non-ground negative literal (i.e. floundering)

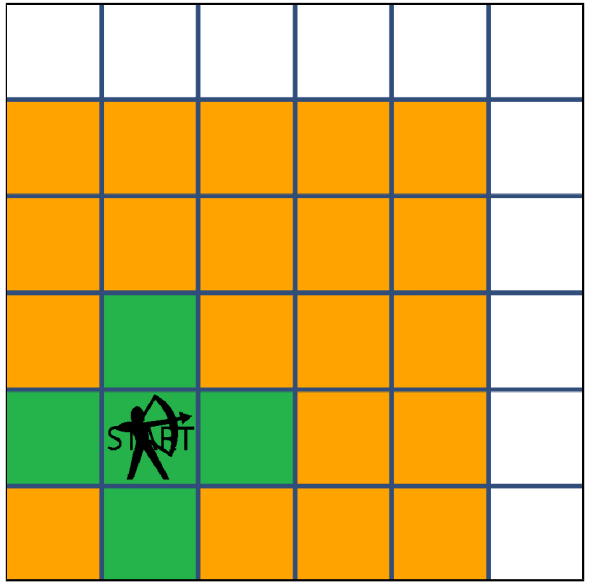
# Global Architecture




Architecture of the integration of a logic programming environment in a game engine

# Prototype: Wumpus World[2]


(a) View of the world




(b) View of the Agent

## Principle of Wumpus World

An agent explores a cave, finds the gold and leaves without dying.

[2] Example extended from Russell and Norvig [2021], and Warren [1999]

# Small Example of a Wumpus World Game Situation

Listing 5: Human agent knowledge of the world

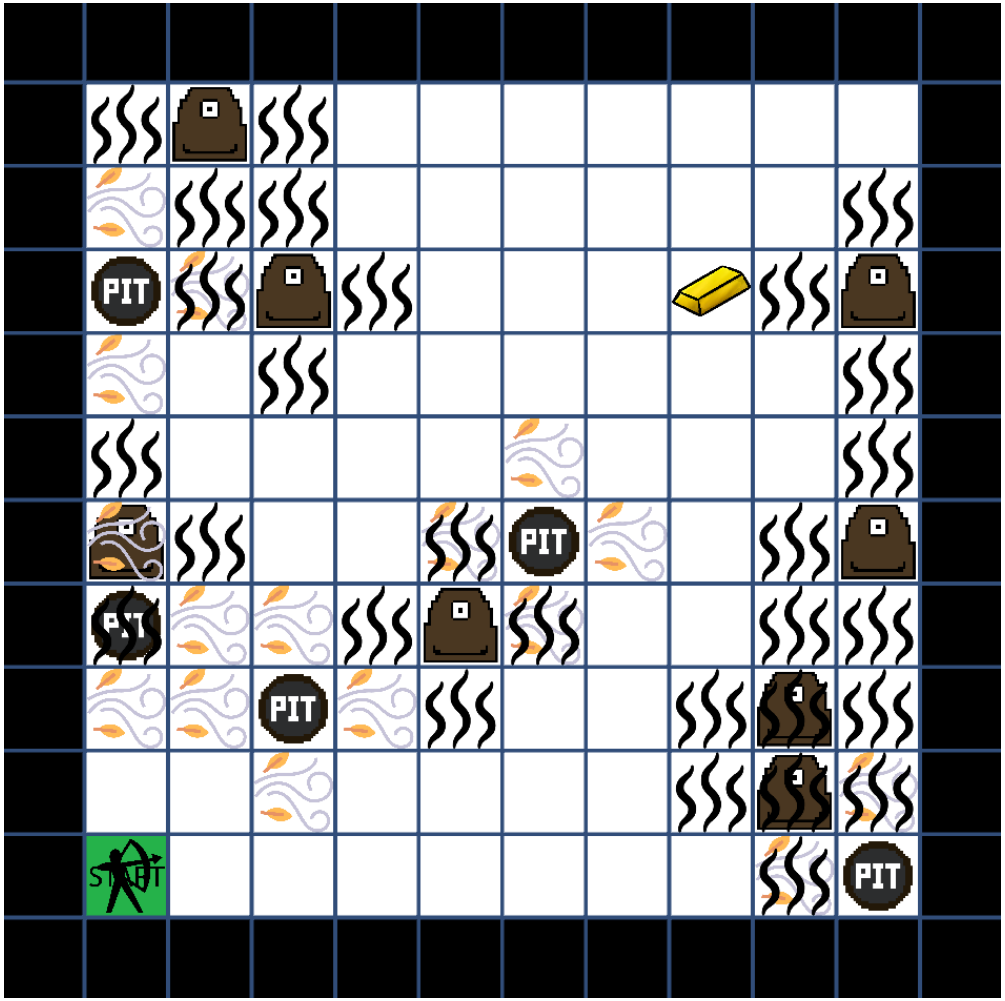
```
visited(1, 1). visited(1, 2). visited(2, 1).
visited(2, 2). stench(2, 2).

safe(X, Y):- tnot(monster(X, Y)).
safe(X, Y):- tnot(pit(X, Y)).
safe(X, Y):- visited(X, Y).
safe(X, Y):- neighbor(X, Y, X2, Y2),
             visited(X2, Y2),
             no_breeze(X2, Y2),
             no_stench(X2, Y2).
```

Listing 6: Prolog query for monster given new knowledge

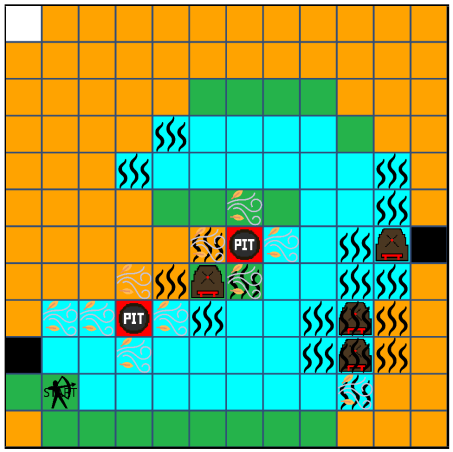
```
?- monster(3, 2).
true.
```

# Adding Complexity



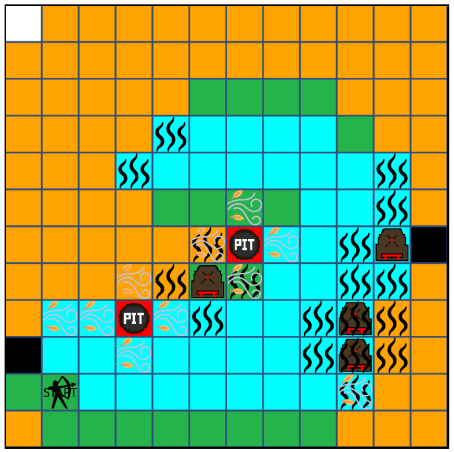

A more complex configuration of the Wumpus World game

# Introducing Behaviours

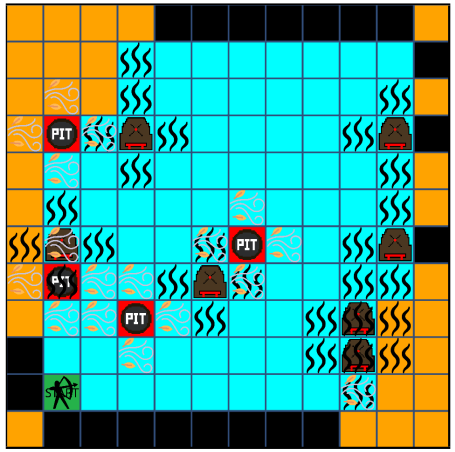



(a) Default behaviour

# Introducing Behaviours

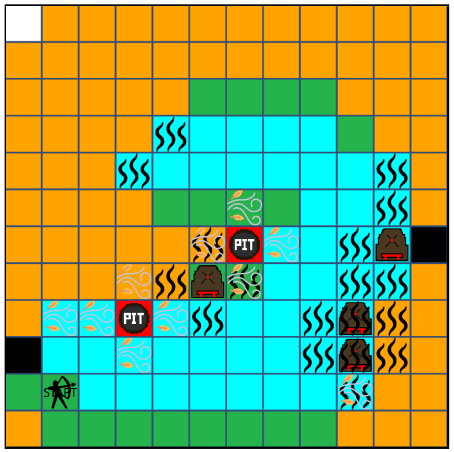



(a) Default behaviour

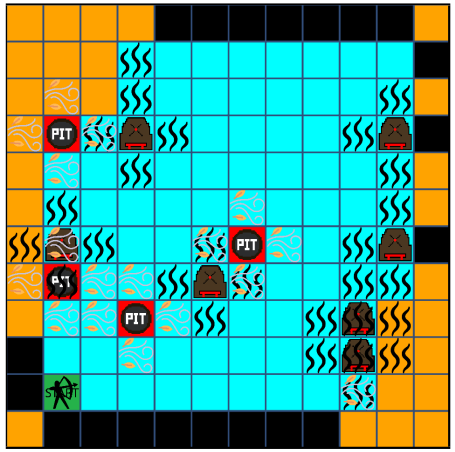



(b) Explorer behaviour

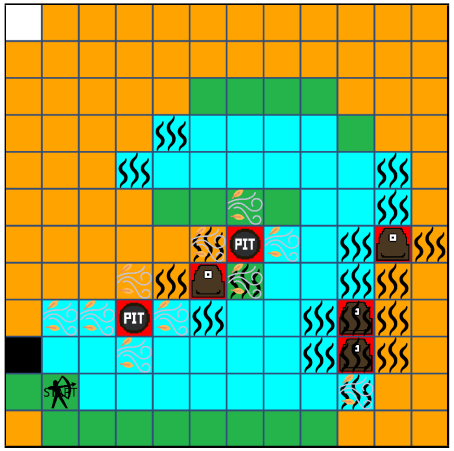
# Introducing Behaviours




(a) Default behaviour

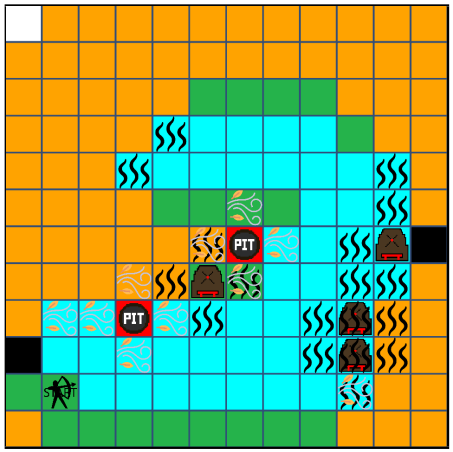



(b) Explorer behaviour

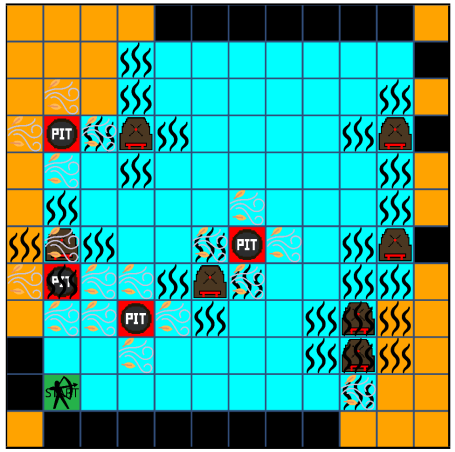



(c) Pacifist behaviour

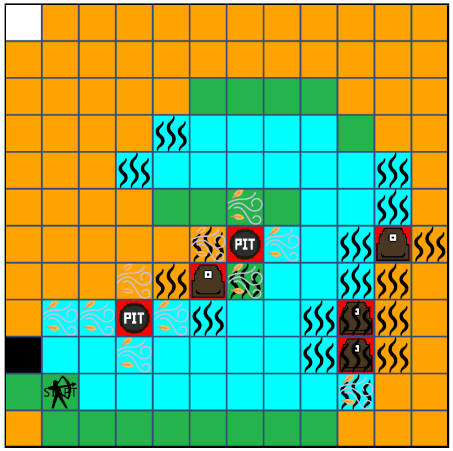
# Introducing Behaviours



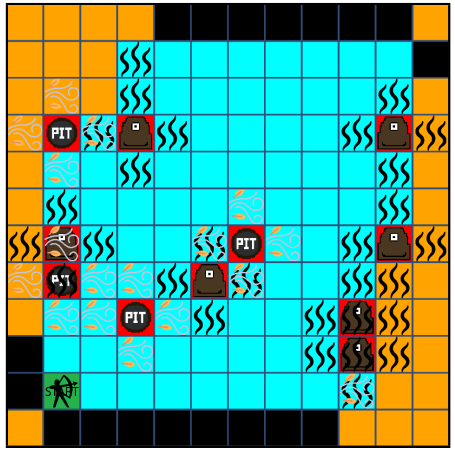

(a) Default behaviour




(b) Explorer behaviour




(c) Pacifist behaviour




(d) Pacifist AND Explorer behaviour

# Current and Future Works

## Current work

- Improve the prototype to enable even more complex behaviour
- Better formalise the creation and interaction with ontologies

# Current and Future Works

## Current work

- Improve the prototype to enable even more complex behaviour
- Better formalise the creation and interaction with ontologies

## Future work

- Integrating and testing AI in Wako Factory's commercial game
- Making Prolog easier to use for game designers outside academia

## Current work

- Improve the prototype to enable even more complex behaviour
- Better formalise the creation and interaction with ontologies

## Future work

- Integrating and testing AI in Wako Factory's commercial game
- Making Prolog easier to use for game designers outside academia

# Thank You!

# References I


Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. pages 1070–1080. MIT Press, 1988.

Ian Millington. *AI for games*. Taylor & Francis, a CRC title, Boca Raton, third edition edition, 2019. ISBN 978-1-138-48397-2.

Stuart J. Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson series in artificial intelligence. Pearson, Hoboken, fourth edition edition, 2021. ISBN 978-0-13-461099-3.

Andrey Simonov, Aleksandr S. Zagarskikh, and Victor Fedorov. Applying behavior characteristics to decision-making process to create believable game ai. *Procedia Computer Science*, 2019.

# References II

Allen Van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):619–649, July 1991. ISSN 0004-5411, 1557-735X. doi: 10.1145/116825.116838. URL https://dl.acm.org/doi/10.1145/116825.116838.

David S. Warren. Programming in tabled prolog, 1999.

Georgios N. Yannakakis and Julian Togelius. *Artificial Intelligence and Games*. Springer International Publishing : Imprint: Springer, Cham, 1st ed. 2018 edition, 2018. ISBN 978-3-319-63519-4. doi: 10.1007/978-3-319-63519-4.