UCLouvain - Université catholique de Louvain

# LINFO1131 Concurrent programming concepts: Projet PacmOz

*Groupe 055:*
Abdullahu Eduart : 03132100
Antúnez García Isaías : 03142100

December 10, 2023

# Contents

# 1 Introduction

For the LINFO1131 course, we were asked to create a program that implements a Pacman game in the programming language Oz. We have designed various agents behaviors, which we will describe in detail. We will also explain the extensions that we have implemented.

# 2 Agents Strategies

The agent strategies that we decided to implement in the game are the PacmOz's and the GhOzt's. These strategies involve different play styles and tactics, and are designed to help the team achieve their objective: Eat all the pacgums or eliminate all enemy players . The PacmOz agents focuse on eating all the pavgums and eating pacpows to elminate GhOzt agents, while the GhOzts focuse on chasing and killing PacmOz agents.

## 2.1 PacmOz Agents

The base version of PacmOz implemants a random move on the maze, it checks all the possible ways from the current position and then it gives a list with the possible directions [ north east west] if the PacmOz was facing north in the previous move (see the examples below)

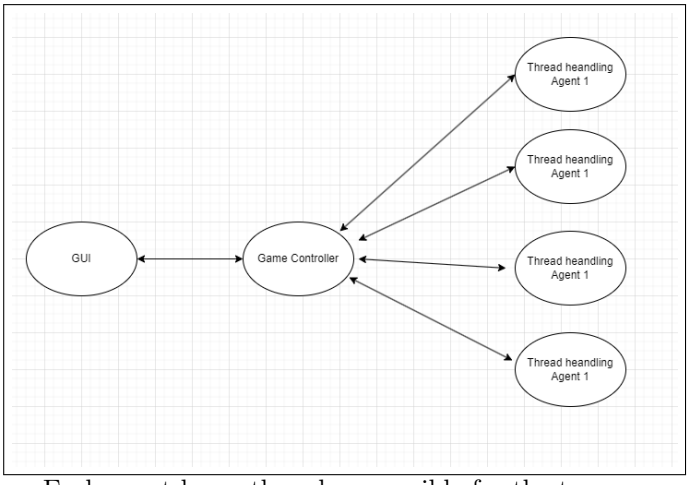
## 2.2 GhOzt Agents

The base version of GhOzt also implemants a random move on the maze, it checks all the possible ways from the current position and then it gives a list with the possible directions [ north east west] if the PacmOz was facing north in the previous move (see the examples below)

# 3 Design of the implementation

## 3.1 Game Controller

The game controller, keep track of every thing happening to the game. It is responsible to for the actions of the agents but also for the different senarios in the game such as maintaining a balance between the agents and their communication.




Each agent has a thread responsible for the turn sequencing, this thread asks the agent what it wants to do at each turn. When it receives a response.
The thread sends a message to the game controller with the information given by the agent.
The Game controller then checks if the agent is able to do the action.
Updates the GUI if needed

### 3.2 GUI

The GUI file was not modified for the basic version only for the extensions explained below.

### 3.3 PacmOz / GhOzt

They both use similar implementation, they have their own state with informations about their id, position, powers, other agents, last move,etc. They also share a lot of functions such as SameBox, RemoveFormRecord etc. These functions helps us to maintain order and also track of agents. Their comunication with the GC is by message handlering sending messages to the GameController and receiving from it. For this impementation we also do previous mouvement track, to not allow Bots to make 180° turns but also have invalid moves like walking on walls, going outside the maze. Both of the agents don't particularly aim to grab pacgums or pacpows, but they do use them if they happen to walk on their tile. But the Advanced versions the use to chase each other a pacmOz chases a GhOzt if they have consumed pacpows and if they dont they chase pacpows, in the other hand GhOzts chase PacmOz if they are powerless and escape from them if they have consumed pacpows.

## 4 Extensions

### 4.1 GUI changes

We made changements on the interface by adding more sprites, and each time the game is launched a randomized sprite is loaded. We added le lava maze, beach maze, jungle maze and urban maze.

### 4.2 AI PacmOz/ GhOzt

A Advanced PacmOz chases pacpows to then use them to kill GhOzt agents, it calculates the shortest distance to the pacpow and than takes the direction thats closest, if two distances are similare ex: north and west it always takes the north , on the other hand it takes the south. Once the power is consumed it chases another on and if there are any pacpows left it makes random moves.

A Advanced GhOzt uses the logic of the official game , chases PacmOz so it could kill it, to achieve it calculates the shortest distance to the pacpow and than takes the direction thats closest, if two distances are similare ex: north and west it always takes the north , on the other hand it takes the south. If a PacmOz takes a pacpow than all the GhOzt make a 180° turn and make random moves like on the official game.

## 5 Conclusion

In conclusion, we had to create/modify a game using only declarative code and message passing between multiple agents. It allowed to practice using ports and GUI programming in Oz, and also forced us to analyze the base structure of the code to find our way around it and be able to make meaningful modifications all while following the guidance of the project statement. The project statement was quite vague (a lot of mistakes on the pdf, a lot of confusion ) which forced us to think outside of the box and learn more about Oz and declarative code in the process.