LoDaLe - team

# Tablut Challenge AI 2024/25

Presentation Slides

# INTRODUCTION

**A PYTHON CLIENT FOR THE ASHTON TABLUT GAME, DEVELOPED FOR THE TABLUT CHALLENGE AT THE UNIVERSITY OF BOLOGNA.**

- Developed entirely on Python
- The best move is selected using tree search  with Alpha-Beta pruning
- Each node is evaluated with a single heuristic function based on 11 different variables
- Terminal state evaluation:
    - White wins: + infinity
    - Black wins: - infinit,
- The timeout check is done at each node's expansion

# GOALS

## SET A NEW RECORD IN TABLUT HISTORY

We wanted to set the bar and be the new standard in the Tablut field

## GET +2 POINTS IN THE EXAM

We can't say it was not a good motivation to start thinking about Tablut

## IMPROVE OUR KNOWLEDGE IN AI, SEARCH ALGORITHMS BLA BLA BLA...

Last and least, we wanted to challenge ourselves in a group project, practicing with the tools that we learned

# CHALLENGES WE FACED

### SEARCH TREE IMPLEMENTATION

After implementing a Node class, we tried to debug our search tree to see if the alpha-beta cuts were correct, so we used our algorithm also on an easier problem.

### NOT CONSIDERING WINNING MOVES

When the agent explored the search tree it was considering only the deepest evaluation, ignoring whenever there was a less deep victory, the result was a player which could not reach the victory, even if we implemented a depth-weighted heuristic.
Spending much more time on this issue would have made us solve it in a better way, but for time constraints we patched it implementing a breadth-first evaluation just for the first layer.

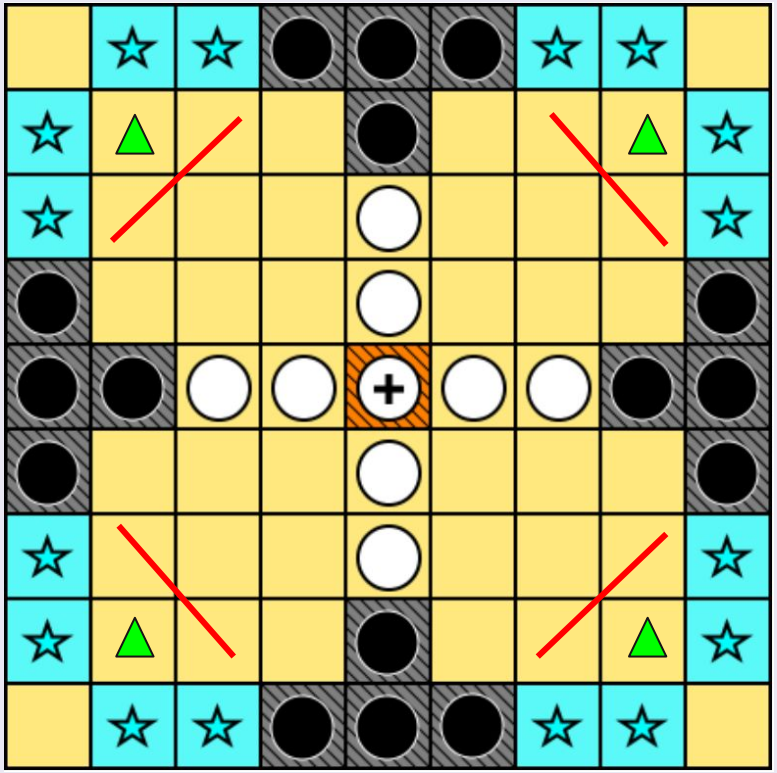
### COMPARING DIFFERENT HEURISTICS

- Games parser
  - Dataset of moves → “Statistical query” attempt :(
- Games executor
- Tradeoff simplicity/accuracy
- Wrote 4 different heuristics and ran all the matches as black and as white, simulating the tablut-challenge

# HEURISTIC

## SINGLE HEURISTIC BASED ON 11 WEIGHT DIFFERENTLY VARIABLES

1. Whites alive
2. Blacks alive
3. Covered escapes by blacks
4. Double covered escapes by blacks ▲
5. Blacks in the rhombus position
6. King surrounding threats
7. King surrounding protection
8. King possible protection
9. King possible checkers
10. King's free route to escape
11. King is in good cells




## ... WE FOCUSED MORE ON SEARCH STRATEGIES

We could have implemented an iterative deepening with transposition table over the alpha-beta cut, in order to guarantee a solution with respect to the available time.

## ... WE HAD TIME TO TEST THE C++ CODE

We could have reached inimaginable search tree depths even with our unnecessary complex heuristic, winning the competition and leaving everyone speechless.

## ... WE PUT HEURISTIC IN A GENETIC ALGORITHM

We could have implemented a genetic algorithm to refine the weights of the heuristic instead of focusing mainly focusing on the heuristic's variables.

# THANK YOU FOR YOUR ATTENTION