ARGUS - A BDI MULTI-AGENT MINECRAFT ENVIRONMENT

by Bardia Parmoun
MASc Electrical and Computer Engineering

COMP 5900

01

INTRODUCTION

MOTIVATION

- Lack of complex multi-agent applications for the BDI design paradigm
- What would BDI-MAS strategies look like?
- What are some successful strategies?

PROBLEM DESCRIPTION

Objectives:

- Autonomous NPCs vs zombies
- Survive the game and collects points

Rules:

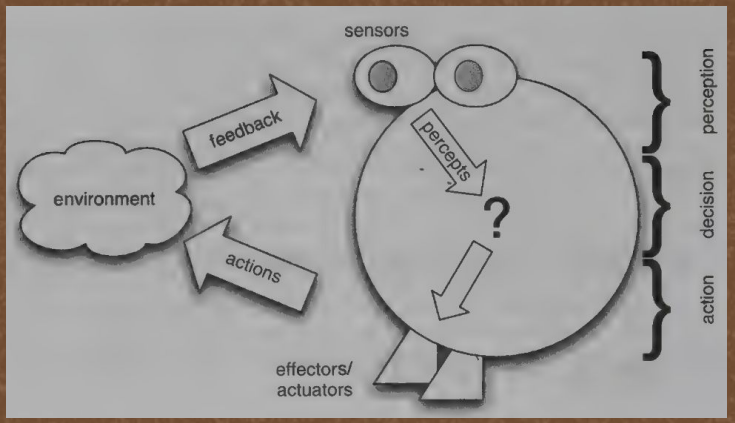
- can attack the zombies and other NPCs
- games run for 1.5 minutes
- can chop trees and collect wood
- can build houses to hide and recover (can fit 2 players at a time)
- can form alliances
  -> share houses, donate wood, protect each other

02

BACKGROUND

WHAT ARE BDI AGENTS?

- "An agent is a **computer system**, situated in some **environment**, that is capable of **flexible autonomous action** in order to meet its **design objectives**"
  - Jennings et al
- Autonomous robots are a great example!
- Popular ways to design robots: imperative programming, state machines, subsumption architecture, etc.
- Proposing **BDI:**
  - **BELIEFS:** agent's knowledge about the world
  - **DESIRES:** the goals for the agent
  - **INTENTIONS:** agent's plans to for achieving its goals



HOW TO WRITE BDI AGENTS?

- **AgentSpeak:** a theoretical formal language for BDI
  - Formal notation for agent's beliefs, goals, and plans

- **Jason:** one of its most popular implementations!
  - The agent repeatedly perceives, reasons, and acts!

Beliefs

Goals

Rules

Plans

```
wallFollowAimAngle(45).
wallFollowDistanceSetpoint(0.5).
!wallFollow(0.8, 32.0).
tooFarFromWall(WallDistance) :-
    wallFollowDistanceSetpoint(SETPOINT) &
    WallDistance > SETPOINT.
tooCloseToWall(WallDistance) :-
    wallFollowDistanceSetpoint(SETPOINT) &
    WallDistance < SETPOINT.
+!wallFollow(WallDistance, WallAngle):
    not(tooFarFromWall(WallDistance)) &
    not(tooCloseToWall(WallDistance))
    <-
    .print("Following the wall...");
    !turn(WallAngle).
+!wallFollow(WallDistance, WallAngle):
    tooFarFromWall(WallDistance) &
    wallFollowAimAngle(WALL_FOLLOW_AIM_ANGLE)
    <-
    .print("Too far from the wall...");
    !turn(-1 * WALL_FOLLOW_AIM_ANGLE +
          WallAngle).
+!wallFollow(WallDistance, WallAngle):
    tooCloseToWall(WallDistance) &
    wallFollowAimAngle(WALL_FOLLOW_AIM_ANGLE)
    <-
    .print("Too close to the wall...");
    !turn(WALL_FOLLOW_AIM_ANGLE + WallAngle).
```

Jason example!

03

THE MODEL

AGENT BELIEFS

Internal Beliefs

Belief	Meaning
lowHealthThreshold(_)	the agent considers this a low health
searchTimeout(_)	how long the agent should conduct a search for
buildRequirement(donation, _)	the agent's donation limit
ally(_)	the agent is an ally with the agent

External Beliefs (Perceptions)

Perception	Meaning
type(_)	the agent type
woodsChopped(_)	the number of woods chopped by the agent
buildRequirement(house, _)	the number of woods needed to build a house
buildRequirement(sword, _)	the number of woods needed to build a sword
buildRequirement(axe, _)	the number of woods needed to build an axe
buildRequirement(trident, _)	the number of woods needed to build a trident
zombieDefenceLimit(_)	the number of zombies the agent can take on at once
allPlayers([_, ..., _])	the list of current players in the game
near(tree)	the agent is near a tree
near(zombie, _)	the agent is near zombies (second parameter indicates the count)
near(player, [_, ..., _])	the agent is near player (second parameter indicates the list of players)
houseCount(_)	the number of houses available to the player
hiding	the agent is hiding in a house
health(_)	the agent's current health (between 0 and 1)
hasWeapon(_)	the agent has a weapon. Parameter indicates the weapon type
damagedBy(_)	the agent took damage from the given entity

AGENT ABILITIES

Desires

Desire	Meaning
!broadcast(_, _)	send a message of given type and content to all players
!sendToGroup(_, _, _)	send a message to a given group of people with a given type and content
!loop	the agent's main logic loop

Messages

Message	Type	Meaning
wantAlliance(_)	askIf	the asking agent wants an alliance
allianceConfirmation(_)	tell	the agent formed an alliance
endAlliance(_)	tell	the agent ends an alliance
hasHouse(_)	tell	the agent announces they have a house
need(_)	askIf	the asking agent needs something
donated(_, _)	tell	the agent donated a number of woods to the agent

Actions

Action	Meaning
say(_)	the agent sends a message to the logs
find(_)	find and navigate to a specific parameter (tree, zombie, [player])
chop_wood	the agent chops a tree
escape	the agent jumps to a random location
attack(_)	the agent attacks a random entity (zombie, [player]) if nearby
build(_)	the agent builds an object (house, sword, axe, and trident)
enter_house	the agent goes to the house
leave_house	the agent leaves the house
donate_wood(_)	the agent donates a certain amount of wood
receive(_, _)	the agent either receives (wood, count) or (house, from)

WEAPONS AND SCORES

WEAPONS	COST (IN WOODS)	# ENTITIES TO ATTACK	ESCAPE RANGE (IN BLOCKS)
NOTHING	0	1	10
SWORD	10	2	8
AXE	15	3	6
TRIDENT	20	4	4

ACTIVITY	COST (IN WOODS)	REWARD (IN WOODS)
BUILD HOUSE	12	500
ATTACK ZOMBIE (PER ATTACK)	0	25
DONATE	2 (DEFAULT)	50
SURVIVE	0	10000/[# PLAYERS]

PROPOSED STRATEGIES

- **Capitalist**
  - try to build as many houses as possible
  - only buy swords for weapons after having at least one house
  - Shares the houses with the allies and donate wood to them
  - Attack zombies
  - Find and attack non allies only if has sword

- **Attacker**
  - try to buy a sword first
  - try to build a house after
  - upgrade weapon to axe and trident only after having at least a house
  - Shares the houses with the allies and donate wood to them
  - Attack zombies and non ally players only if has a weapon

WHICH STRATEGY?
HOW MANY ALLIES ARE SUITABLE?

- Have 4 attackers, 4 capitalists, and 8 zombies

- Three experiments each run 30 rounds:
  - Have agents form no alliances
  - Have attackers and capitalists pair up to alliances of size 2
  - Have attackers and capitalists team up to two teams of size 4

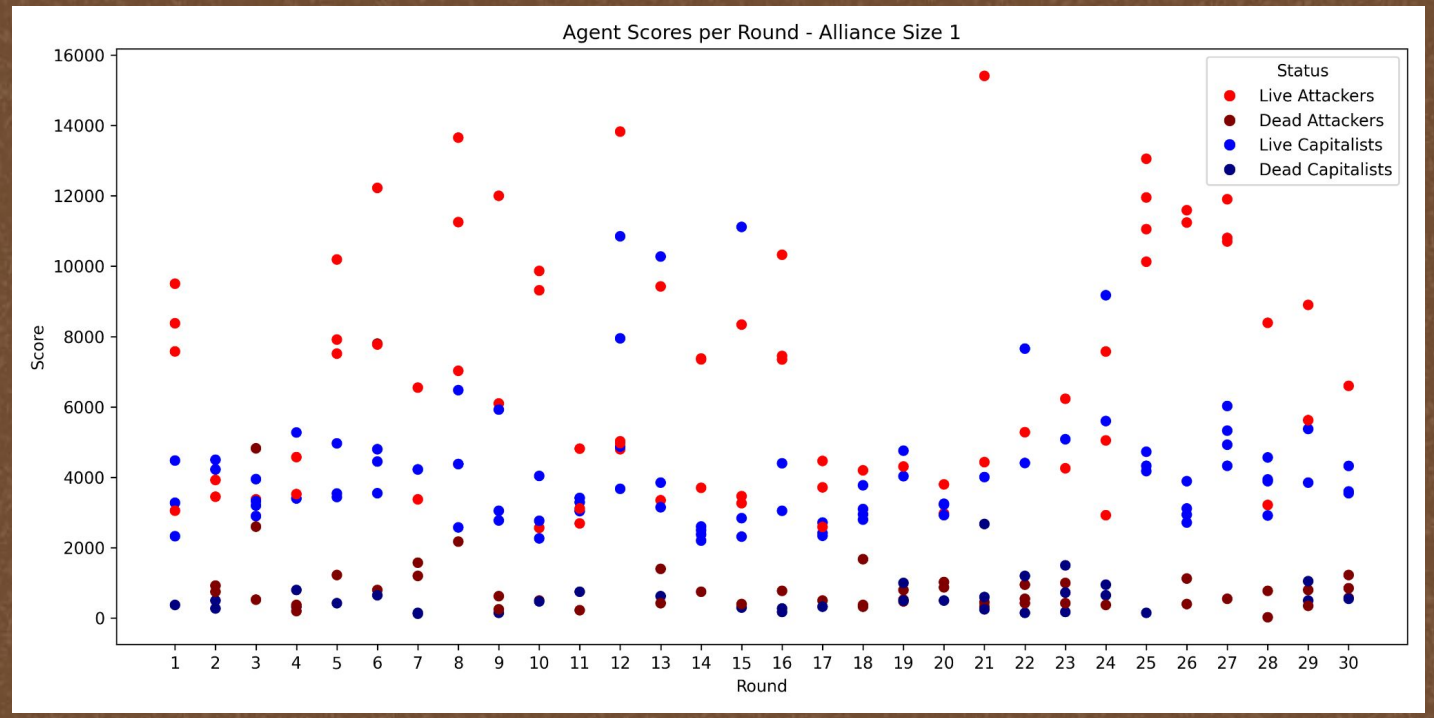
04

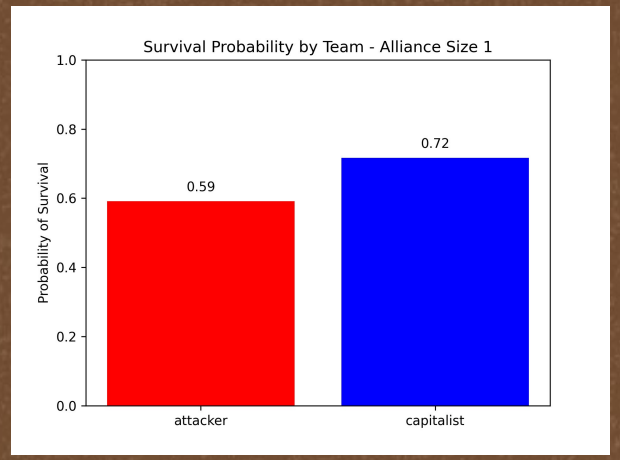
DEMO!

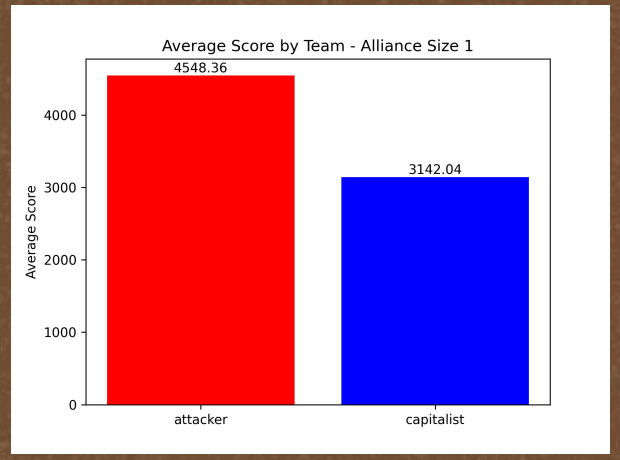
05

RESULTS

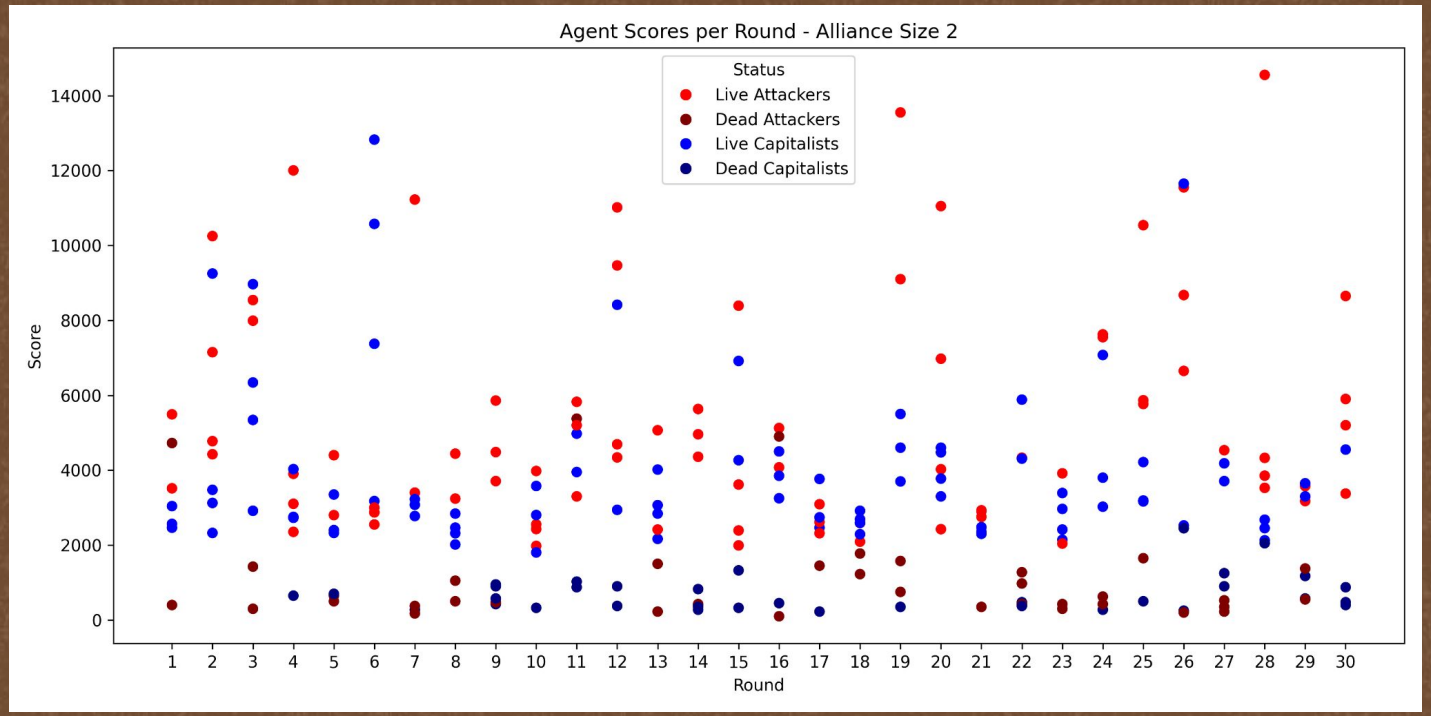
ALLIANCE SIZE 1

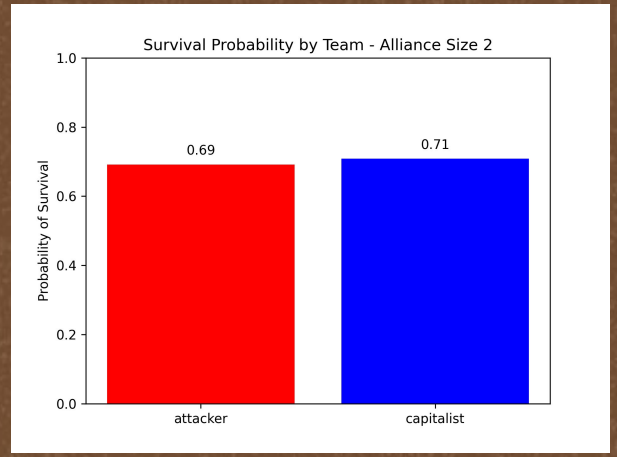


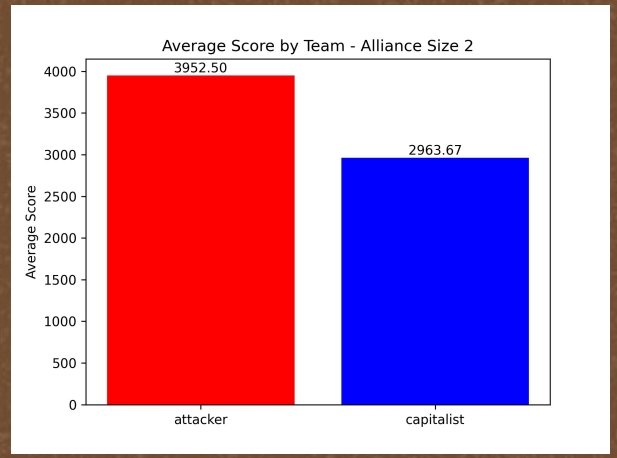




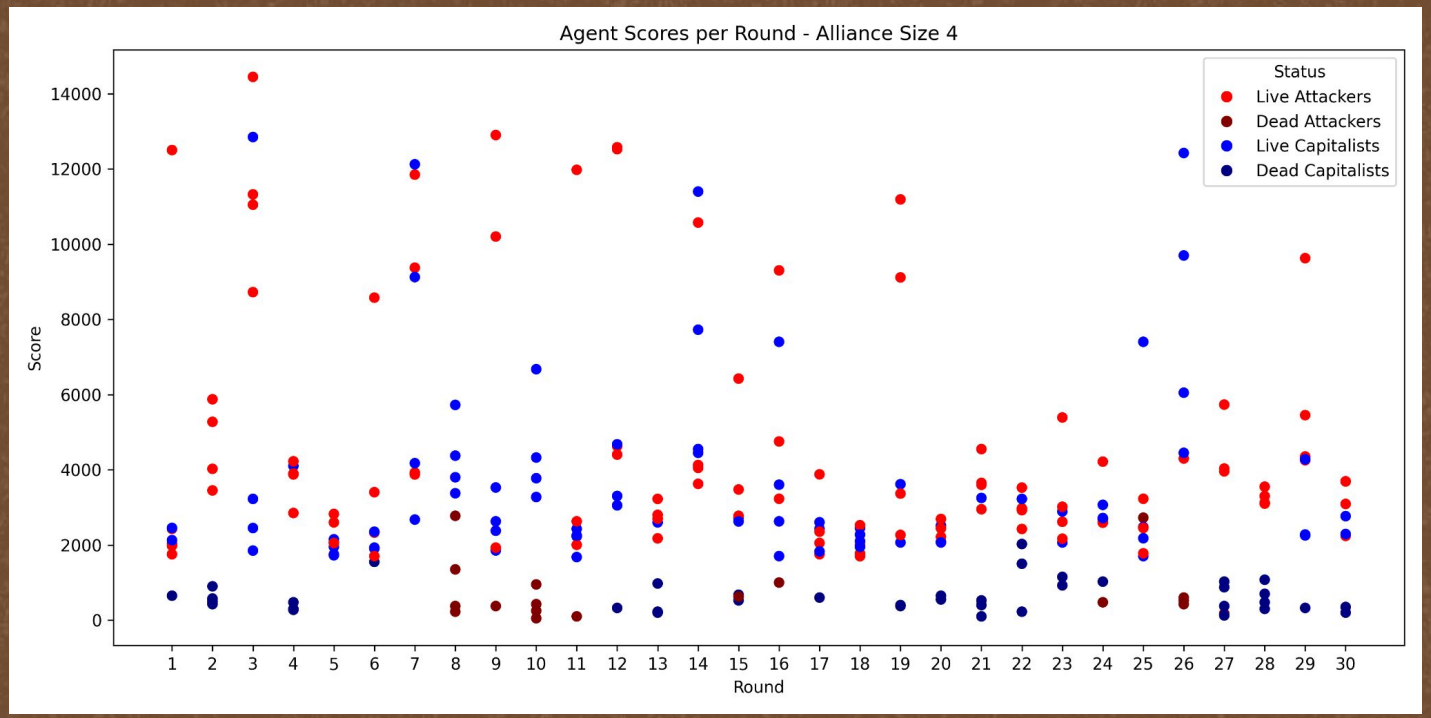
ALLIANCE SIZE 2

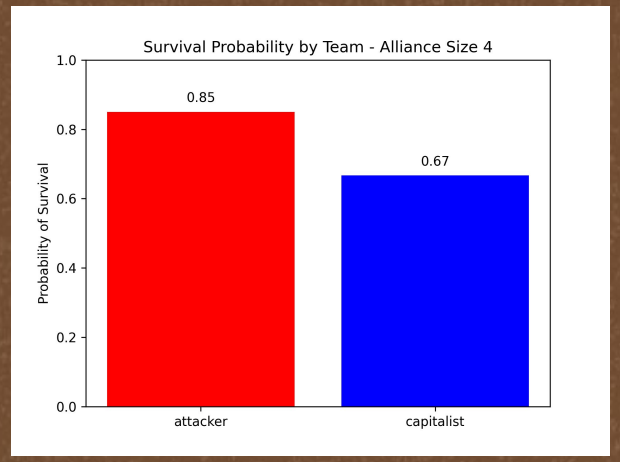


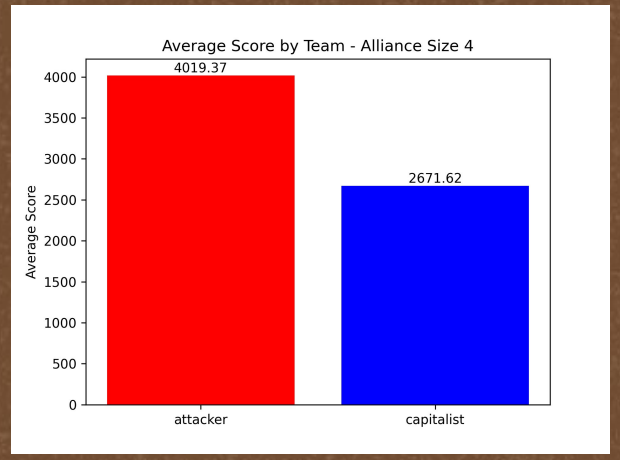




ALLIANCE SIZE 4







06

CONCLUSION

PROBLEM DESCRIPTION

Overall Findings:

- Jason is able to keep up with the fast paced (20 ticks/second) environment
- AgentSpeak makes agent's implementation clear and transparent
- BDI leads to easy belief maintenance and plan selection

Strategy Findings:

- Attackers are generally more successful -> invest in a lot of weapons
- Being an attacker requires more allies though!
- For small alliances, agents need to invest in houses but also attack a lot!

FUTURE WORK

MORE COMPLEX STRATEGIES

MORE EXPERIMENTS/DATA COLLECTION

MORE COMPLEX FEATURES

COMPRE OTHER PARADIGMS

THANK YOU