Project - Distributed AI & Intelligent Agents

Perttu Jääskeläinen Group 1 Gabriele Morello
perttuj@kth.se morello@kth.se

January 2, 2023

Contents

1 Introduction

In this project assignment, we created a simulated festival with different types of guests that move around and interact with other guests, stages and various locations based on their personal preferences. Each guest has a set of personal preferences depending on their type (with some randomness included). Guests interact with each other using the FIPA protocol, and decide upon their actions using BDI (belief, desires, intentions).

As an extra effort, we implemented BDI and reinforcement learning to alter guests behavior based on environmental conditions, such as an introverted guest avoiding extroverts and larger crowds.

2 Approach - Problem formulation

This section covers the problem formulation - i.e. what are we trying to solve or implement in this task on a higher level.

To ensure we cover all the requirements, we started the project with constructing a certain set of requirements that should be implemented. Using inspiration from earlier homeworks as a basis, we decided to implement a festival simulation with different guest types instead of learning a new platform or working on an entirely new scenario.

Below we discuss how we approached each of the requirements for the project.

- **Create at least 5 different types of guests**

To approach this, we started off with the two most simple guest types - **Extroverts** and **Introverts**. We also introduced the types **Security Guard**, **Addict** and **Fighter**, which themselves have varying amounts of extroversion/introversion.

- **Each guest type has at least 1 different set of rules on how they interact with other types**

Building on the guests referred to in the previous section, we constructed an association matrix that impacts a guests utility for performing actions based on what types of other guests are nearby that location. If the utility is greatly reduced by the other guests at each location, the guest in question will change their course of action.

For example, if an **Introvert** guest wanted to attend a stage with utility 10, their utility might get reduced to 2 because of very many extroverts also present at the stage - so they might choose their second option, which has utility 8 and very few other guests attending.

	Introvert	Extrovert	Security Guard	Addict	Fighter
Introvert	1	0.2	0.4	0.4	0.2
Extrovert	0.2	1	0.6	0.4	0.2
Security Guard	0.6	0.8	1	0.2	0.4
Addict	0.8	0.4	0.4	1	0.2
Fighter	0.8	0.4	0.2	0.6	1

In addition to these values impacting guests utility, guests will also occasionally interact with other guests in the following ways randomly when they are near each other:

	Introvert	Extrovert	Security Guard	Addict	Fighter
Introvert	Small Talk	Avoid	-	-	-
Extrovert	Offer Drink	Take Shots	-	-	Avoid
Security Guard	-	-	-	Send to jail	Send to jail
Addict	-	-	Avoid	-	Avoid
Fighter	-	-	Annoy	Fight	-

- **They also have at least 3 personal traits that affect these rules**

We used the following personal traits for all guests, with varying values. See the table below for exact counts, which impact the interactions between guests. Each trait has a value between 0 and 1, and may impact the choices guests make when interacting with others.

- **Sociability**: if a guest prefers being by themselves (less than 0 value), or if they are fine with more people (high value). This applies to locations such as stores or open spaces, i.e. not music stages where guests might be dancing.
- **Energy Density**: if a guest prefers many guests at the same location enjoying music for themselves, perhaps being loud or very energetic. For example, an introvert might have a low value here, since they prefer a more controlled and low-energy environment.
- **Spontaneity**: if a guest prefers to get out of their comfort zone and attempt something new to spice things up. For example, an introvert might occasionally want to force themselves to try something new, so they might have a high spontaneity value even though they might usually prefer more quiet environments.

The table below specifies the values for each guest - where the values may range from -1 to 1.

	Sociability	Energy Density	Spontaneity
Introvert	0.1	-0.5	0.2
Extrovert	0.8	1	0.8
Security Guard	0.2	0.8	0.1
Addict	-0.5	1	0.2
Fighter	0.2	0.3	0.1

- **They have at least 2 different types of places where agents can meet**

Using inspiration from the previous assignments, we included **Stages** and **lounge area**, to where agents may navigate depending on their preferences.

- **Use at least 50 guests in your simulation/Make simulation continuously running**

We ran the simulation with different guest amounts to experiment with how agents would end up interacting.

- **Agent communication with FIPA for long distance messaging**

We implemented long distance communication using BDI, and FIPA for normal interactions as described above (for example, offering a drink is done over FIPA rather than direct communication with agents).

- **Have at least 1 global and interesting value to monitor and display on a chart**

To do this, we measured the **fun** that guests have at the festival - which was dependent on how many times they encountered annoying/disturbing guests, which would reduce their fun during the time they were at the festival.

We also tweaked the "cost" of encountering such a guest, to see how it impacted the overall experience of guests.

- **At least 1 useful and informative graph**

We created a graph for the social links between guests - both before and after implementing reinforcement learning - to see how guests would group up together.

- **Draw out at least 1 interesting conclusion from the created simulation**

We didn't have any plan for what type of conclusion to attempt to draw from the simulation at the beginning of the project, but decided on monitoring the value of **fun** that guests have at a festival depending on the "cost" of encountering annoying/disturbing guests being an interesting value to look at.

We also planned to monitor how reinforced learning impacts the same value, if or when we implemented it.

3 Implementation Details

The code is structured such that we have a base class, called **festival_guest**, which has all the shared behavior of all species, such as attributes and generic actions/reflexes. For example - navigating to different stages, moving to the chill area to relax and going to jail in case security catches a guest being too drunk or violent.

The main behavior that's interesting in the **festival_guest** is the **listen_music**, **choose_closest_stage** and **return_to_base** plans, which define how the guest will behave at any given moment (in addition to a few others). We chose to make the base implementation using BDI to avoid duplicate work (one implementation with vs. one without). See the next chapter about details in our BDI implementation.

Each sub-species has its own implementation depending on the interactions it has with other agents. For example, the **introvert** sub-species (which has **festival_guest** as its parent), implements the **avoid** and **smallTalk** actions, which interact with **extroverts** and **introverts** respectively.

We implemented the species as defined above, i.e. the following (with different styles):

- **introvert**: colored as **green**, implements the **avoid** and **smallTalk** interactions.

- **extrovert**: colored as **yellow**, implements the **offerDrink** and **offerShots** interactions.
- **security**: colored as **orange**, implements the **sendToJail** interaction.
- **addict**: colored as **blue**, implements the **avoid** interaction.
- **fighter**: colored as **red**, implements the **annoy** and **fight** interactions.

We also have some more insignificant species, that are also important for the functionality of the festival. These are the **stage**, **lounge** and **jail** species. We also have an additional species, **socialLinkRepresentation**, to display relationships between guests.

- **jail**: represents a square that guests will go to when interrupted by a security guard. The square is colored in **gray**.
- **lounge**: represents a square that the guests will go to relax when they've been dancing for long, or when they want to avoid other guests. This is colored in **black**.
- **stage**: represents a triangle which the guests will attempt to find, such that they can enjoy themselves to some music. Is colored in yellow with a black border.
- **socialLinkRepresentation**: used in our graphs to display relationships between guests. See figure 6 for details about what these look like.

See figure 1 for an example of what all of these agents look like when running the simulation.

4 Beliefs, Desires, Intentions

To satisfy the first extra challenge we implemented most interactions using the Belief-desire-intention (BDI) software model. This architecture provides a way to separate the phase of selecting a plan from the phase of execution. The three main components are Beliefs, Desires, and Intentions:

- **Belief** represents the information known to the agent it includes factual information and abstract information, they may not be true that's why they are called beliefs
- **Desire** represents the goals and objectives that the agent has, the agent has also to believe that the goal is achievable
- **Intention** represents the actions that the agent plans to take in order to achieve its desires. they must compatible with the goals.

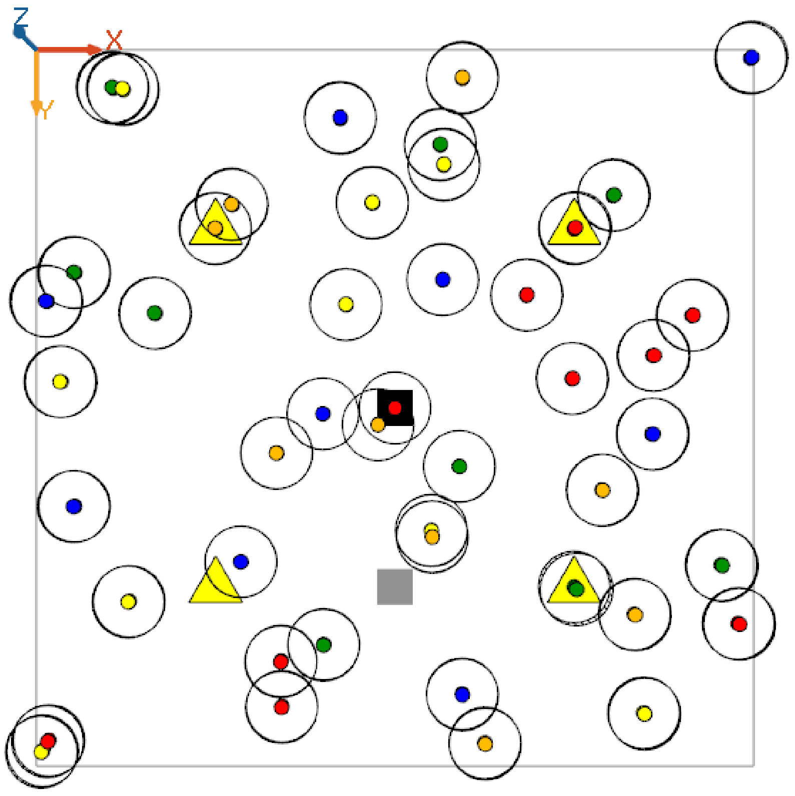


Figure 1: *An example run of guests with their individual colors.*

BDI agents are a built-in feature in GAMA, to operate with BDI in Gama we have to define the global predicates, define species perceptions, rules and plans.

- **Perception** is executed at each iteration and operates on beliefs. We used perceptions to detect other agents to share information and compute the likeability.
- **Rule** is a function that is applied at each iteration to generate new desires or beliefs based on the agent's current set of desires and beliefs. We used rules to add the intention of resting when the agent is tired and to add the desire to be free when arrested.
- **Plan** the agent has a collection of behaviors called plans that are designed to achieve specific goals. Some plans are, for example, listening to music, resting and choosing a stage

We also used BDI to manage social relationships, Gama provides several attributes: liking, dominance, solidarity, familiarity, and trust. We used liking to determine whether an agent shares his information with other agents, we define likeability as the distance between agents' musical tastes, every agent has in fact an array of preferences. The likeability is a real value between -1 and 1.

5 Reinforcement Learning

To solve the second challenge we added to each guest a memory, we used it to store an expected fun for each stage, the expected fun is computed every time a guest visits a stage to listen some music, when he arrives we save the initial fun and when he is tired and he is leaving we subtract the initial fun to the current one and we combine it with the old average fun that the agent had in that particular stage.

When an agent has to decide where to go he choose the best stage based on the expected fun, the choice is a weighted random choice whose weights are the expected fun values fro each stage. Initially the expected fun for never visited stages is very high but the initial value does not take part in the average after the first visit.

6 Experimentation and Results

We ran the simulation with various settings - such as not having any interactions between guests at all, and running with different agent counts/property values.

We can see the differences between values of fun in figures 5, which shows the fun value in a basic run (without any sub-species), and in figure 3, which shows the fun when introducing all sub-species. In the initial figures, we noticed that we have a *worse value* when using sub-species interactions - which kind of makes sense, since a lot of the interactions have a negative impact.

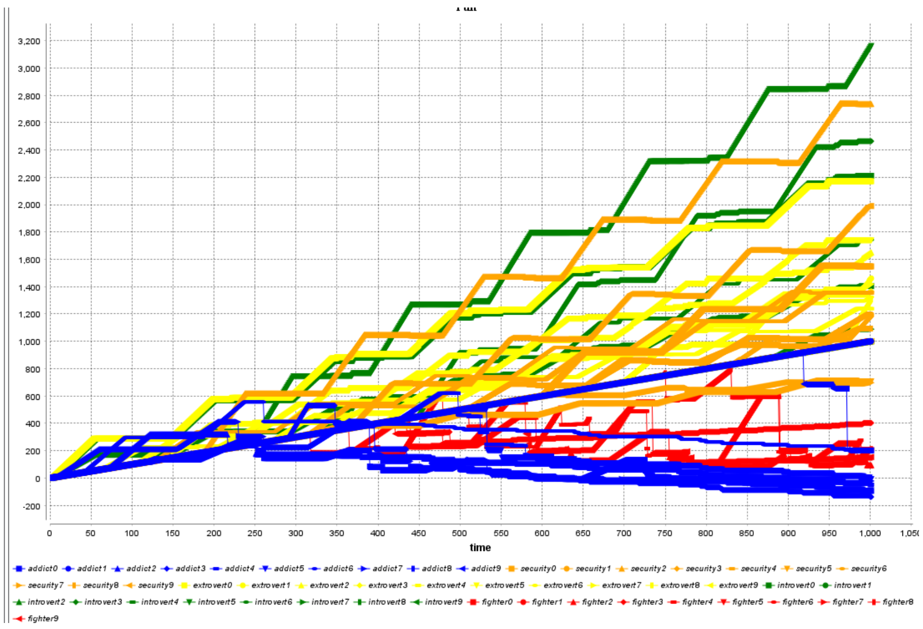


Figure 2: *A graph of fun values between different festival guests after 1000 cycles with all rules in use (i.e. all species and their interactions).*

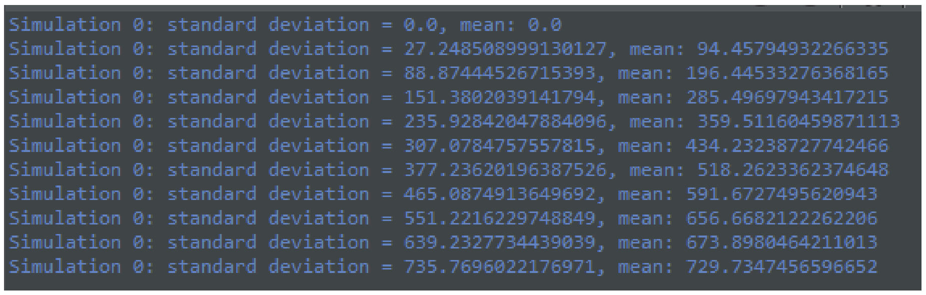
We can also see the links between the agents in figure 6, which doesn't use reinforcement learning so agents are spread out across the festival.

7 Discussion and Conclusions

In this project, we could go more in-depth into how agents communicate and act based on their intentions using BDI. We also learned more in depth about reinforcement learning, and how agents can dynamically adapt to their environment to improve their behavior.

The conclusion we can draw from this is that the potential of the GAMA platform is huge, where agents can be implemented as if they were people - where they can perceive their environment around them, and act as if they were people. We can also construct beliefs, desires and intentions very clearly, which is an intuitive and easy thing to implement clear behavior in agents.

We also conclude that having interactions between agents can greatly impact how they enjoy their time at the festival - having just some negative value between guest interactions can very quickly cause certain guests to have a bad experience - but avoiding negative interactions can instead greatly increase the fun guests are having as well.



```
Simulation 0: standard deviation = 0.0, mean: 0.0
Simulation 0: standard deviation = 27.248508999130127, mean: 94.45794932266335
Simulation 0: standard deviation = 88.87444526715393, mean: 196.44533276368165
Simulation 0: standard deviation = 151.3802039141794, mean: 285.49697943417215
Simulation 0: standard deviation = 235.92842047884096, mean: 359.51160459871113
Simulation 0: standard deviation = 307.0784757557815, mean: 434.23238727742466
Simulation 0: standard deviation = 377.23620196387526, mean: 518.2623362374648
Simulation 0: standard deviation = 465.0874913649692, mean: 591.6727495620943
Simulation 0: standard deviation = 551.2216229748849, mean: 656.6682122262206
Simulation 0: standard deviation = 639.2327734439039, mean: 673.8980464211013
Simulation 0: standard deviation = 735.7696022176971, mean: 729.7347456596652
```

Figure 3: *Standard deviation and mean value for fun with all species in use (i.e. all interactions).*

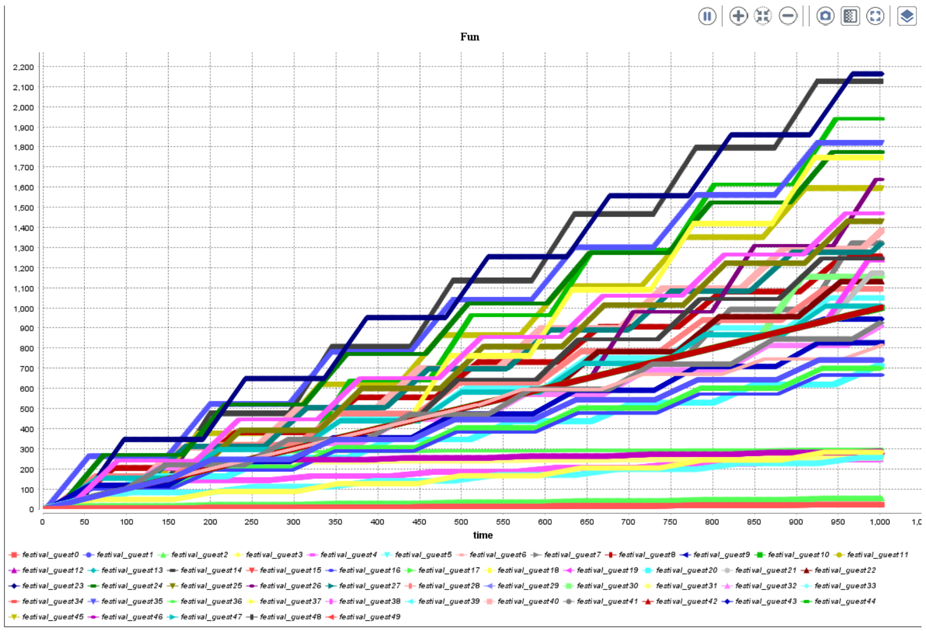
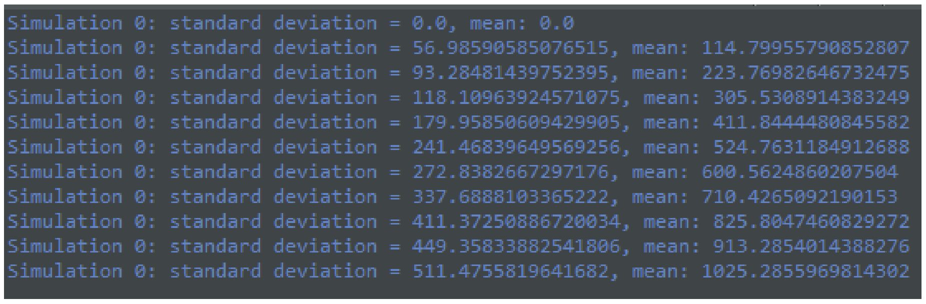


Figure 4: *A graph of fun values between different festival guests after 1000 cycles without any interactions.*



```
Simulation 0: standard deviation = 0.0, mean: 0.0
Simulation 0: standard deviation = 56.98590585076515, mean: 114.79955790852807
Simulation 0: standard deviation = 93.28481439752395, mean: 223.76982646732475
Simulation 0: standard deviation = 118.10963924571075, mean: 305.5308914383249
Simulation 0: standard deviation = 179.95850609429905, mean: 411.8444480845582
Simulation 0: standard deviation = 241.46839649569256, mean: 524.7631184912688
Simulation 0: standard deviation = 272.8382667297176, mean: 600.5624860207504
Simulation 0: standard deviation = 337.6888103365222, mean: 710.4265092190153
Simulation 0: standard deviation = 411.37250886720034, mean: 825.8047460829272
Simulation 0: standard deviation = 449.35833882541806, mean: 913.2854014388276
Simulation 0: standard deviation = 511.4755819641682, mean: 1025.2855969814302
```

Figure 5: *Standard deviation and mean value for fun with only basic guests in use (no interactions).*

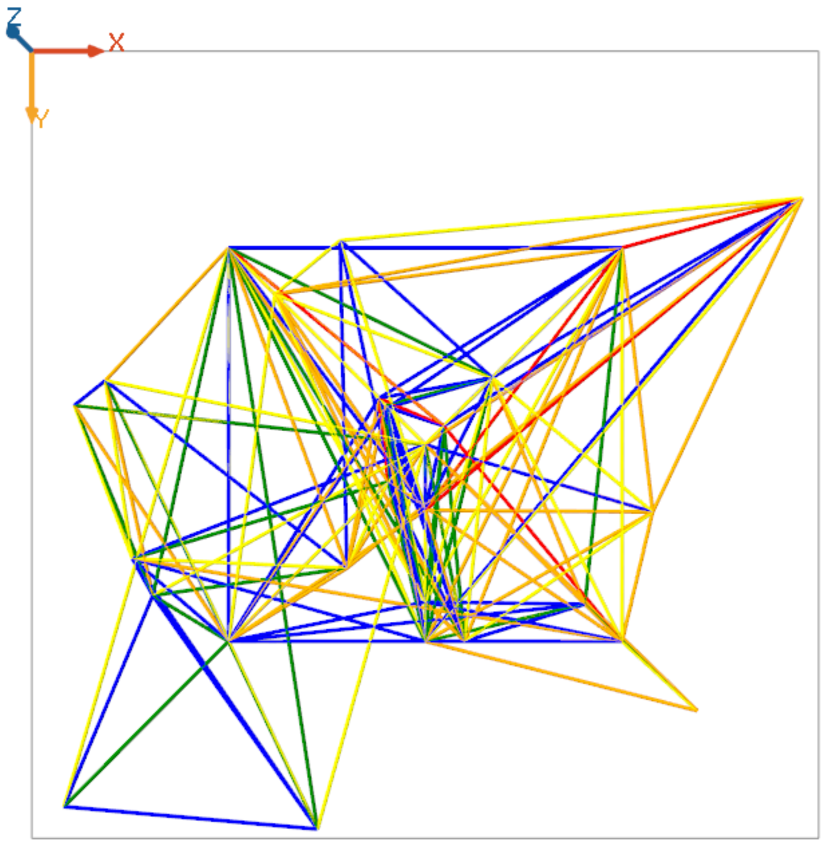


Figure 6: *Social links between guests, where colors determine different ranges of values in relationships between guests (without reinforcement learning implemented).*