# Base Part

In this final project we were asked to simulate an environment where different type of agents communicate and interact with each other using GAML and the GAMA Platform.

In the basic task we had to implement at least five different types of agents, using at least 50 agents in our scenario. The agents have at least one different set of rules on how they interact with other types, and 3 personal traits that affect these rules. The agents will be able to meet in two different type of places and will communicate entirely with the FIPA protocol. The simulation can be continuously run.

## How to run

1. Run GAMA 1.8.0
2. Import all the files in the folder “base”
3. Press the green button “Festival” in top left corner to start the simulation

# Species

## Guest

The species guess is by far the most complex in our environment since it represents the actual attendees of our festival simulation, and it’s where most of the conversations between agents are happening. All the code can be found inside the Guest.gaml file.

Inside a Guest we defined different traits:

* *chill2dace*: this is the trait that we use to indicate whether an agent wants to dance or to chill. It’s implemented as a float value between 0 and 1. When this vaule is greater than the *danceTrashold* the Guest will go to the Dance floor, otherwise it will go to the Chill zone. *chill2dace* increases over time with the reflex *updateChill2Dance* and can also decrease by talking to other agents.
* *Thirsty* is a value used to check if the agent wants to go to the Bar and drink. This value updates through the reflex *updateThirsty*, which increases the value by a small random amount
* *Talkative* is used to indicate how much a Guest is willing to talk to other agents and it is used in multiple reflexes where an agent is trying to start a conversation
* *Love:* this variable represents how much an agent wants to meet a soulmate, and it is used together with *loveTrashold*, *gender* and *desiredLoveMateGender* in reflexes that take place inside the Tinder Area.
* *Drunkness* is used to check how much alcohol a Guest drunk during the simulation. If *drunkness* exceedes a certain value the guest is kicked out from the festival by the Security Guard.

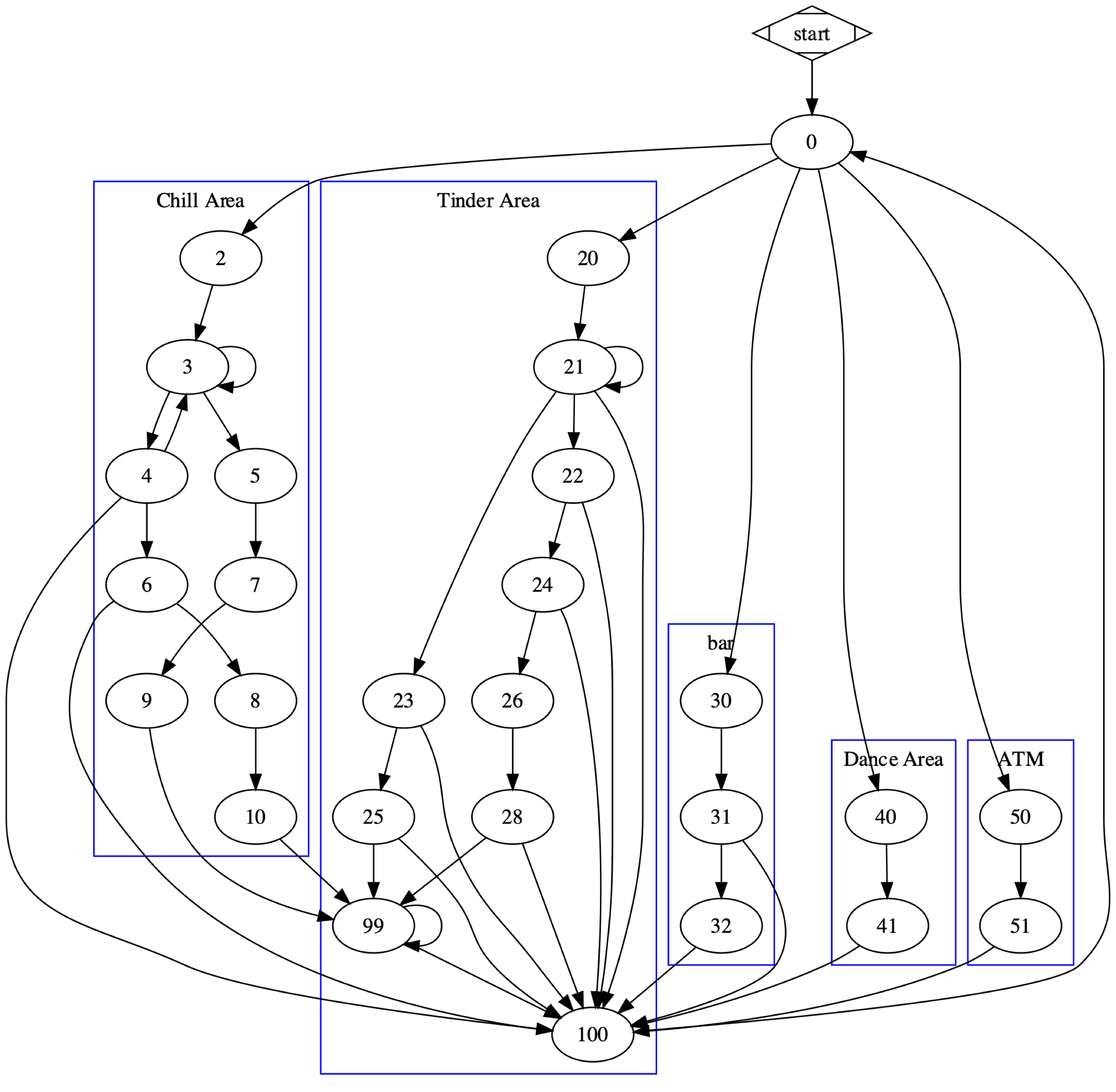
Other variables are used for other parts of our simulation and will be described if and when necessary.

Our guests are divided in either **Chill** or **Party**: the first one will tend to prefer the Chill Area and its *chill2dance* parameter will increase much slower compared to the Party Guest. The *talkative* is also higher in a party guest.

To better represent these two types of agents we wrote two distinct species, *ChillGuest* and *PartyGuest*, both children of the specie *Guest*.

## Finite State Machine

The entire Guest specie functions as a Finite State Machine, where each state represent a specific situation that can or will happen during our simulation. This choice was made because it guarantees a robust implementation of our agents, while still keeping the code relatively clear to understand.



The **State 0** is the first state in which our agents are “spawned” in. Here each agent will evaluate its own traits in relation to the various thresholds and go, thanks to reflexes such as *goToChillArea*, *goToTinderArea*, *goToBarLocation* and *goToDanceArea*, to the area that best fit the Guest parameters, changing the status from 0 to a new one.

**State 100** is used to reset each agent to the state 0 if deemed necessary and it represents the end of all the conversations with other type of agents.

## Chill Area States

When an agent’s *chill2dance* is below its own *danceTrashold*, that Guest’s status is set to **2**, and with *goToChillArea* the guest will set its own targetPoint to the *ChillLocation*. Once arrived, *arrivedAtChillArea* is triggered and the status is set to **3**. The Guest is now executing *randomWaitChillArea*, where the agent is wandering while looking for potential conversations with other Guests. A conversation is not always initiated thatks to the *flip* statement that prevents Guests with a low *talkative* to talk as much as others. If the *flip* returns true, the agent will go to **State** **4** and start looking for potential neighbours to talk to in the reflex *startConverationChill*. Once a partner is found, the first agent will be the initiator of the conversation, and it will go to **State** **6** and communicate the table chosen to go to talk, more on this later. Other agents in the Chill Area that are still in status 3 will keep executing *catchConversationStart*, a reflex design to listen to potential initiators and begin new conversation if approached. If a guest is approached it will go to **State 5** and send a confirmation to the agent that started talking. The reflex *gotACKfromChillGues* triggers and sets the initiator’s destination to the chosen table, once there the status will be set to **8** with *initiatoreReachedTableChill*. With *targetGoesToTableChill* the approached will also go to the same table as the other Guest and set its status to **7**, once reached the destination the status will become **9**. Now that both agents are at the table, the initiator will communicate its own *chill2dance* and go to **State 10**, all in the reflex *sendC2dToTarget*: this recreates a situation where the approacher is trying to convince the other guest to either go chill or dancing. The receiver now executes *gotC2D*, where a new *chill2dance* is calculated using the logic that we developed. Now that the exchange has been terminated, both agents go to **State 99**, a state often used in our simulation where the agents just wait a number of cycles (*WAITING\_ITERATIONS*) to better represent a real-world conversation that could be happening between two people.

## Tinder Area States

The second area where the Guests can meet is the Tinder Area, called like these because here the agents try to approach other guests based on what gender they are attracted to. Once the *love* trait of a guest goes above the *loveTrashold*, that agents will go to the Tinder Area and set its status to **20**, reached the destination the status will be set to **21**. At State 21 the agent will execute *lookAround*, where with the same mechanism that involves *talkative* previously described will be used to look for a partner. If the *flip* returns true, the agent will go to **State 22** and execute *lookingForSoulMate*. The algorithm works similarly to the one developed for the Chill Area, the only difference being that here we have more messages and states because we have to communicate each gender to check if it’s what the approacher is looking for. Once 2 agents agree that they can talk, they will go to the booked table and start talking, exchanging the *chill2dance* parameter the same way as before.

## Bar States

When the *thirsty* parameter’s threshold is reached,