Negotiation and Communication (FIPA)

ID2209 Distributed Artificial Intelligence and Intelligent Agents

Group 19
Emil Ståhl
Peyman Peirovifar

Negotiation and Communication (FIPA) Assignment II

Emil Ståhl and Peyman Peirovifar

November 14, 2020

1. Introduction

This report covers the implementation of a multi agent-based festival simulation written in the modeling language GAMA. In this version of the festival simulation we extend our simulation with a new type of agents, auctioneers. The auctioneer will pop up at least once during the simulation and sell merge to the winner of the auction. The communication between the agents is done thru the FIPA protocol. Auctioneer starts his offer with much higher price than the expected market value. If no one wants to buy for the set price, he reduces the price at a selected interval. The auctioneer decides how much he reduces the price in every round. If the price is reduced below the auctioneers minimum value, the auction is cancelled. The goal of this work is to get more experience working with Agents in GAMA and an introduction to message passing and the FIPA protocol.

2. The program and its purposes

The program consists of two types of agents/species (similar to classes in objective oriented programming). These agents are guests and auctioneers.

3. Implementation

Much of the code from our previous work "GAMA and Agents" is reused in this code base, for example are the Guest species almost identical to the ones used in the previous simulation. However, the Guest species is extended with a new reflex called receive_inform_messages. This reflex is responsible for handling the bids from the auctioneers. Each guest has one item that they are going to bid for at a highest acceptable price. The reflex checks if the current item that the auctioneer is taking bids on is equal to that one the guest is interested in. If the guest is not interested a "refuse message" is sent back to the auctioneer.

```
do refuse with: [ message :: m, contents :: ['not-interested'] ];
```

If however the guest is interested in the item and wants to make a bid it will send a "agree message" to the auctioneer.

```
do agree with: [ message :: m, contents :: ['interested'] ];
```

The guest will also calculate its new bid which increases with each round that it is going to send to the auctioneer. Finally, a propose message is sent to the auctioneer with the item and price.

```
do propose with: [ message :: m, contents :: [productID, acceptablePrice] ];
```

Now to the new species called auctioneer. The first reflex is the start auction which is executed each minute. A list is created with all guests that are within a specified range from the auctioneer, these are the possible participants of the auction. If there are no possible participants the auction is halted. A item from the merchandise list is chosen and a price is set. Lastly, the reflex informs all possible participants that a specific item is being auctioned at a specific price.

```
do start\_conversation with: [to :: possibleParticipants, protocol ::

'fipa-contract-net', performative :: 'inform', contents ::  [productID, price] ];
```

Next reflex receive_refuse_messages is handling the refuse messages from the participants. The reflex iterates thru the refuses and ends the conversation with the specific agent. The number of participants are also decremented. The most important reflex is the receive_propose_messages that handles the actual bidding. It loops over all the proposes and for each proposal it checks if the proposed price is higher than the lowest acceptable price. If so the item is sold and an accept proposal message is sent with the following structure.

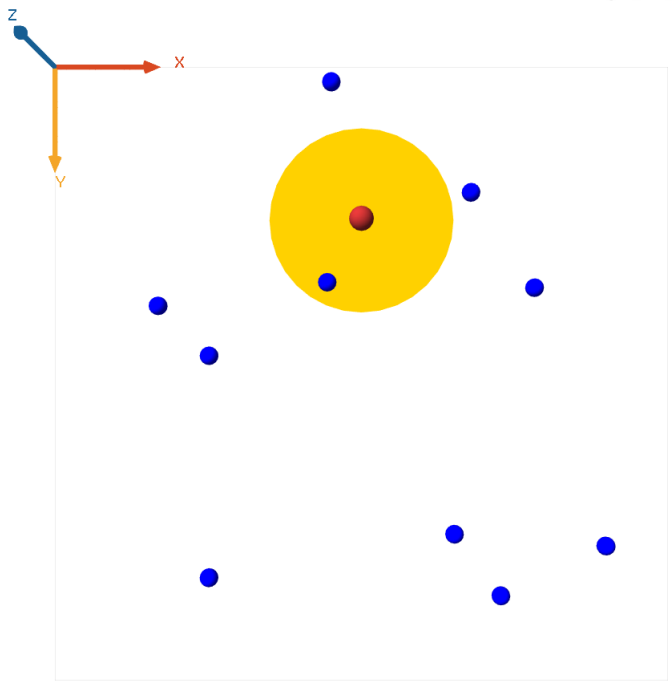
```
do accept\_proposal with: [ message :: m, contents :: [] ];
```

If the price is lower than the auctioneer rejects the proposal. If the item is not sold during the round a new price is calculated and communicated to the participants.

4. Results

Shown below is part of the message log and a snapshot of the simulation running.

```
(Time 56760.0): Auctioneer0 initiates an auction for CD.
The initial price is 814 euros.
Guest0 proposed 3886 euros.
Guest0 bought CD for 3886 euros.
The auction by Auctioneer0 is over.
(Time 56820.0): Auctioneer0 initiates an auction for T-shirt.
The initial price is 735 euros.
Guest0 is not interested.
Guest(0) left the auction
(Time 56880.0): Auctioneer0 initiates an auction for iPhone cover.
The initial price is 983 euros.
Guest0 is not interested.
Guest(0) left the auction
(Time 56940.0): Auctioneer0 initiates an auction for CD.
The initial price is 887 euros.
Guest0 proposed 4274 euros.
Guest0 bought CD for 4274 euros.
The auction by Auctioneer0 is over.
(Time 57120.0): Auctioneer0 initiates an auction for T-shirt.
The initial price is 564 euros.
Guest0 is not interested.
Guest(0) left the auction
```



5. Conclusions

This work gave more experience working with Agents in GAMA and an introduction to message passing and the FIPA protocol. The most challenging part was to figure out how to structure the auction.