Multi-Agent systems - Assignment 6 (Competition)

Emma Machielse, Federico Tavella, Alessandro Tezza

March 18, 2018

1 Introduction

The current version has a competition strategy implemented, where buses bid for the allocation of newly added buses to their area.

2 Bidding

Our transportation system is divided into four areas that each have a bus assigned as a coordinator for this area. These four coordinators ("local coordinators"), have the responsibility to check whether the amount of passengers waiting in their area does not rise above a certain threshold. One of these coordinators, the "global coordinator", also performs this check for the four areas altogether. This global coordinator creates new buses based on this global check if necessary.
This is when the bidding happens. When a global coordinator creates a new bus, it sends a message of type "*bid-request*" to the other local coordinators. They check the number of people waiting at the bus station in their area, and send this number as a bid to the global coordinator. The global coordinator checks which bid is the highest, i.e. which local coordinator is in most need of a new bus. It assigns the new bus to the area of the local coordinator with the highest bid.
When a new bus is created, it has to wait until it gets a message that specifies the area in which the bus should go. This message is sent by the global coordinator: after having chosen the winner of the auction, the global coordinator sends a message to the newly created bus with the schedule that it has to follow.

3 Other improvements

- When no passengers are waiting, all buses stop travelling, to reduce travelling costs
- We are currently refactoring the code: the agents.nls file is splitted into 5 files. The agents.nls code describes the core BSI framework, linked to action execution. Then there is an init file, that states how the first buses are created and initialised. The execute-intentions file describes the action functions needed for each intention. The messages file incorporates some utility functions to work with the messages. Finally we have a utils file that contains all the functions needed by the rest of the application. In the refactored version, submitted this week, the competition is implemented. However, some work still need to be done in order to implement all the feature of the past versions (that is the reason of the decrease in terms of performance of this version of the algorithm).