ID2209–Distributed Artificial Intelligence and Intelligent Agents

Assignment 3 - Coordination & Utility

Group 11
Alexander Carlsson - alecarls@kth.se
Abyel Tesfay - abyel@kth.se

2021-11-26

Introduction

In this assignment we have managed two different tasks. First we handled the n queens problem where we made queens communicate and move on a chessboard so no casualties were present. Second we managed a utility task where guests at a festival are to choose what act to go to based on a utility calculated based on guest preferences and how act attributes match these. This has made us more comfortable with FIPA protocols as well as given us practical experience with utility functions.

How to run

Run GAMA 1.7 (or newer). Create a new project and import the files from the zip. Select one of the files to view the implementations of the base tasks

Species

Task 1

Queen

The Queen is an agent that will try to position itself on a chessboard such that no two queens share the same row, column and diagonal line. To coordinate this, each Queen will communicate with its predecessor and successor Queens. If a Queen has no available position, she will tell her predecessor to reposition, so that she can find a position that does not break the conditions.

Task 2

Stage

An agent that hosts different acts for a fixed time, Guest agents can travel to a Stage to view the current act. Stage agents have a set of attributes with different utility values e.g. sound, band, light shows, visuals etc. When a new act starts, the values of the attributes change and the Stage sends the attribute values to all Guests through the FIPA protocol.

Guest

An agent that travels to the Stage agents to view their current act. Guests keep track of the utility of current acts and if the current act ends or if a new act with higher utility appears the guest moves to that stage instead.

Implementation

We started with development on task1 where we chose to make a basic implementation with many visuals in order to allow for better debugging. Once we had a working version which managed 4x4 chess boards in good time, we started to look at complexity as 8x8 and higher problems were not solved in reasonable time. We went from a brute force solution to a smarter solution by observing that only one queen would be present per row. The final solution which solved our complexity issues instead used a row-based brute force function. Now queens go in order by their id, get assigned to the row of their id and try the first allowed position on that row. If the next queen is not able to find any suitable position, the earlier queen tries a new available position in her row.

Results

The two tasks resulted in one program able to solve the N queen problem according to all deliverables in a very short time. The second task resulted in a program where stages change acts at random times with new attributes and guests who move between stages based on available utility at all stages.

Discussion/Conclusion

Task 1 was a bit harder with requirements for more new ways of working compared to other assignments. Managing task 2 was easier as it was more of what we already knew and we were able to get a working version working in a shorter time. Through the events encountered during this assignment we believe that we now have a better understanding of unity functions and further comfort with FIPA message handling. We have also widened our understanding of GAMA in general and are ready to take on the coming project.