Report 1 : Inception

Luca Deltodesco, Matthias Hueser,
Andras Slemmer, Cliff Sun, Luke Tomlin

October 26, 2011

1 Key Requirements

1. AI is capable of playing an entire game of Deplomacy from start to finish.
2. AI is capable of negotiating with other players in the specified negotiation language.
3. Software is capable of handling and organising a game of Deplomacy involving multiple players/AIs.

2 Extensions

1. AI is capable of beating other less complicated AI (eg. ones that do not negotiate, with simple negotiation tactics etc.).
2. Software is extendable to allow other AIs to play (eg. Daide).
3. Software is runnable on multiple platforms.
4. AI is able to negotiate in a meaningful way, based on previous gameplay within a game (Machine learning?).

3 Choice of Development Method

Methodology Agile development, regular group meetings and weekly re-adjustments of targets. Individual targets and group targets set regularly.

Code testing Unit testing and other code testing using QuickCheck.

Version control Handled by Git, using GitHub as a central repository.

Required technology

Server Haskell : Concurrent programming, Network Programming

Socket programming (for GUI/Server communications)

GUI Haxe : GUI creation, cross-platform availability

AI Haskell

Project Management

Progress After the initial server and game creation is completed, progress on the AI can be made.
AI progress should be fairly simple to gauge, as it gains negotiation functions and performs better against other opponents (human/AI).
Additionally, the visuals and interactivity of the game itself will become more advanced and user-friendly as the GUI improves. (eg. from just a simple CLI -> GUI/CLI -> GUI -> GUI with prettier graphics...)

Roles and responsibilities Roles are quite flexible as there are many ways to organise group members within the project, and there are many different jobs that require doing.

Meeting and scheduling arrangements Weekly or more frequent brief meetings to discuss progress. Online forums for other discussions on work and to suggest reading materials.

4 Draft Schedule

Weeks 1 - 2 Write the basic game foundation, CLI interfaces, functions to make it play nicely with external AI (eg. from Daide). Learn how to use the various technologies required.

Weeks 3 - 4 Start initial work on the AI. Continue to improve and develop game foundation (work will need to be split between team members here). AI should be able to play a 'holding' game and fulfill all requirements to play with other AIs. Begin research into tactics and how we will develop the AI.

Weeks 5-6 Further improvements on the AI, implementing extensions and investigating cross-platform availability. Frequent tests against other AIs, test implementations of different AI techniques (neural networks etc.)

Weeks 7+ Finishing touches to game and platforms. Focused work on AI improvement/features.

5 Detailed Plan for First Iteration

Extensible maps

Will require

Representation

- A graph of visitable nodes, each node representing a location (sea/coastal/bicoastal/inland).
- An association of nodes to supply depots or standard location.
- An association of nodes to players starting areas.

Considerations: Spain in standard map has two locations on one territory which represent different nodes in the graph, although they are considered a single territory which effect the movement of fleets.

GUI

- An image representing the map.
- A set of coordinates corresponding to the position of a node on the image.
- Also perhaps require more information to display fleets/armies in different positions on a territory.
- For convoy, perhaps require more information regarding path through the seas to display a nice representation of the convoy actions, if we ever do anything animated also etc.
- Graph will require different arcs - some for Fleets, some for Armies, some for Convoy.
- This will be used to decide if a given move is valid.

Server

Will require

- Method for allowing players to connect (whether AI or human).
- Method for dealing with disconnections (saving their game state, allocating new thread on rejoin, default moves whilst absent?)
- Method for notifying players of changing game stages (start, negotiation, move submission, resolution).
- Method for allowing players to communicate with each other (privately, or as a group).
  - Server does not need to understand the language of negotiation (as defined in Daide).
  - All the server needs to do is relay the message.
- Method for allowing players to submit their moves (with possible added functionality for checking if a move is valid).
  - This will need to be in a game-defined language.
  - Using the language specified by Daide would allow us to 'plugin' their own AIs.
  - Method for parsing this 'move submission' language will be required.
- Method for resolving moves. This includes

  - Moving units location on map (standard move, convoy orders).
  - Resolving disruptions of a move/convoy action.
  - Resolving attack moves and defend moves (and support moves).
  - Notifying a player of the need to retreat/withdraw if they're displaced from a location.
  - Default moves for when move is invalid/not received.
  - Dealing with specific/rare cases.
- Updating and redistributing the current game state.

GUI

Will require

- Method for representing map and units on the map in a clear and understandable fashion.
- Method for submitting orders (CLI -> GUI + CLI -> GUI?).
- Method for allowing players to view a send messages to other players (via the server).

Other notes

Assuming that we can get all of these up and running (with some simplifications to move submission/negotation/map representation), we should have a working 'game', with which we can play (as humans) or 'plugin' an external AI from Daide. Thinking will need to be alloted to how the individual parts communicate (sockets? some protocol?) and how the external AI talk with the server.

Once this is done, we can split the group if necessary, one to focus on improving the general game platform, and another to focus on building the AI to play (and hopefully beat) the other players!

**** Remember that the ultimate goal of this project is to create an AI that negotiates in a game of deplomacy. All of this extra stuff just makes it look a lot more impressive and fleshes out the project.****