Project Proposal

Introduction:

This project proposes to utilise reinforcement learning techniques within the area of AI Game Creation and Selection. Working within the domain of card games, it proposes the creation of two AI agents in order to achieve a system capable of generating new, unique card games and ensuring the games are logically infallable.

Reinforcement learning is a subfield of artificial intelligence regarding the "computational approach to understanding and automating goal-directed learning and decision-making" (Richard S. Sutton, 1998, p. 28). Recently, projects such as AlphaGo by (DeepMind, 2017) and OpenAI's venture into professional E-Sports (OpenAI, 2017) have brought reinforcement learning into the public eye, using the domain of competitive games as an environment to teach complicated AI agents.

In particular, multi-agent reinforcement learning is applicable in these domains due to their ability to "produce behaviors that are far more complex than the environment itself" (Bansal, 2017, p. 1), allowing for interesting applications to be explored. One goal of this project is to utilise such behaviors, generated in this case from two agents; one being reliant on the other as a co-operative agent in the learning process, and the other acting as its own competitve multi-agent system for simulating opponents.

Both of these processes have independently been investigated with regards to their efficacy in improving the learning abilities or performance of AI agents. With reference to the co-operative process, Tan (1993) suggests "agents engaging in partnership can significantly outperform independent agents" (p. 1). Furthermore, Bansal (2017) offers support of "self-play" in reinforcement learning, attributing effectiveness to "agents of comparable strength provid[ing] the right challenge … facilitating maximally rapid learning and avoiding getting stuck" (p. 1). Incorporation of both these practises into a unified system has been researched in less detail, and as such is one of the areas of consideration in this project.

Creativity is an unavoidable talking point when describing the proposed project. Computational creativity, defined by *The Association for Computational Creativity*, is "the study and simulation, by computational means, of behaviour … which would, if observed by humans, be deemed creative" (2016). Card games are traditionally human-made and creating new ones can easily be thought of as a creative task, causing this project to inherently span into the realm of computational creativity.

The extent of how 'creative' any system will truly be is a question pertaining to many different advancements within this field, with some experts "question[ing] the extent to which AI can develop its own sense of creativity" (IBM, 2017). By studying the generated behaviors of the AI system proposed in this project, this question can be further examined within the previously underexplored domain of game creation. Determining the creativity of such a system may ultimately end up being a subjective metric, as IBM's John Smith asserts "we still have to define what creativity means" (IBM, 2017), and as such is not an explicit objective of the project. Nonetheless, exploring this angle of evaluation is intriguing as percieved creativity could be a potentially beneficial trait in this domain.

In summary, the rationale for the proposed project centers around investigating the use of different reinforcement learning methodologies in multiple AI agents. By using these different methodologies in confluence within a limited environment, the output of such a system can be analysed with regards to the complexity and eventual successfullness of the card games generated. Further consideration will be given to the level of computational creativity percieved in the system.

Aim:

The aim of this project will be to create a multi-agent system of artificial intelligences trained using reinforcement learning techniques and cooperative methods. These two agents, "Player" and "Architect", are detailed below.

The aim for the endpoint of the system will be the output of a valid 'GamePlan' object. A GamePlan object will contain elements necessary for understanding and playing a card game, including the rules, goals and win conditions. During the reinforcement learning process, such an object will be generated by Agent B, and validated by Agent A.

Agent A: Player

This agent is focused around the ability to play card games. A GamePlan object will be provided to Player, and Player will attempt to play the game against versions of itself. The actual 'skill' required by Player is minimal, perhaps even playing randomly, so long as the moves it makes conform to the presented set of rules and goals within the GamePlan.

After an attempted runthrough, Player will return a set of heuristics about the game which was attempted. Importantly, this will include a metric determining whether the given GamePlan is viable to play through, effectively allowing Player to act as a classifier for Architect (Agent B). It will also return metrics garnered from the runthrough in order to provide more data to Architect to help optimise the learning process.

Agent B: Architect

This agent is focused around the ability to generate GamePlans. Architect will begin with a simple relational map of key concepts for game creation such as the possible win conditions and rules.

Based on the iterative results of Player, weighting will be applied to this map in order for future iterations to learn and improve upon. In this way, conglomerations of rules that cause the game to be unplayable will be systematically weighted down until they no longer appear in the generated GamePlans. A possible extension from this point would be to use the heuristics from Player to also favour certain subsets of rules or properties of card games, allowing Architect to generate more complex or computationally creative behaviors.

Objectives:

1) Instantiate basic data structures and code framework.
   -Basic elements of the system will be defined and a file structure for code files will be implimented. Examples include 'card' and 'deck' objects to be used in Rule or Goal creation, as seen in the next step.

2) Create a repository of all possible card game 'Rules' and 'Goals' available within the system.
   -The data structure used to compose a card game. Rules and Goals will contain data relevant to how cards are played and how a game is won.

3) Instantiate the GamePlan object type, primarily composed of a subset of rules and goals.
   -This data structure is critical to the system. Instances of this object will be the input to the Player AI agent, and the output of the Architecht agent.

4) Impliment Player.
   -An AI agent capable of interpreiting GamePlan objects into a set of instructions, and using them to simulate many iterations of playing card games against itself.

5) Test Player.
   -A thorough testing strategy must be deployed before proceeding further, as the heuristics being returned by Player directly affect the development of Architect.

6) Impliment Architect.
-An AI agent capable of using reinforcement learning techniques to eventually be able to generate new GamePlan objects when given certain input data.

7) Use Player to teach Architect.
-Use the output of Player as the input data for Architect in the learning process. The outcome of this will be creating a 'brain' for Architect.

8) Have Architect produce new GamePlans.
-Once trained, Architect will be able to generate new, logically consistent card games in the form of GamePlan objects.

9) Test integrity of Architect's GamePlans.
-These will primarily be validated using Player. Further checking can be performed manually; although far less extensive, human testing by phyiscally attempting to play the card games can provide a usefu final evaluation.

Academic Literature:

Reinforcement Learning: An Introduction (Richard S. Sutton, 1998)

This work covers a range of key concepts, algorithms and applications within the field of reinforcement learning. It comprehensively explains terms which persist throughout many papers on the topic, providing a solid entry point for research around this project. The author, Richard Sutton, has made significant contributions to the field and "is considered one of the founding fathers of modern computational reinforcement learning" (Wikipedia, 2017). Although very comprehensive in scope, the generality of the book means some topics necessary to this project are not included, particularly with regards to multiple agent approaches.

Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents (Tan, 1993)

Delving further into research regarding multi-agent systems, this paper establishes their effectiveness over single agents in certain domains when applied to many reinforcement learning problems. Cooperation between learning systems is an overarching theme of both the paper and the project, providing helpful information about responsive learning methods between agents. However, the paper focuses more heavily on the effects on learning rate than the behaviors resulting from this process, somewhat limiting the usefulness of the paper's findings in this area.

Markov Games As A Framework for Multi-Agent Reinforcement Learning (Littman, 1994)

Although Markov Decision Processes (MDPs) were touched upon in the first piece of literature (Richard S. Sutton, 1998), this paper extends the use of MDPs into a multi-agent environment with contrasting agent goals. This is particularly pertinent within this project when relating to the design of Architect and the possible decisions it can make when interacting with Player during the learning process. The paper deals with a fairly restrictive class of environment, and provides insight into how this project may tackle the similarly restrictive environment of card games.

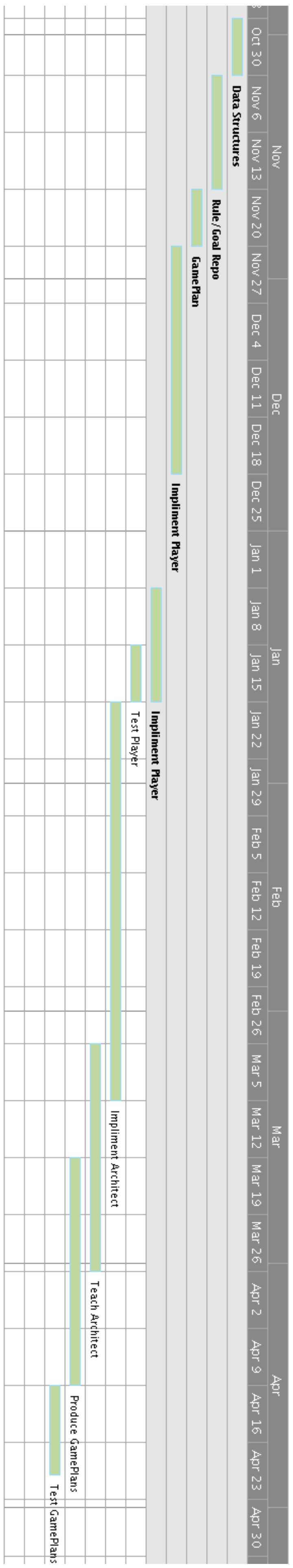
Emergent Complexity via Multi-Agent Competition (Bansal, 2017)

Generating complex behaviors from a relatively simplistic environment is the focus of this paper, especially relating to an agent learning via interaction with simulated iterations of itself. The environment described in this work involves agents exploring a 3D interactive world, which is a much more complex framework than the Markov games previously described by Littman, and extends beyond the scope of this project. Nevertheless, the behaviors generated by these agents is allegorical to those emergent in the proposed project, and the paper provides useful insight into topics such as a learning curriculums and appropriating difficultly level.

Mastering the game of Go with deep neural networks and tree search (Silver, 2016)

The seminal paper regarding the AlphaGo team's achievement in defeating human experts at the game of Go. Reinforcement learning and neural networks are used in conjunction to map decisions through an incredibly complex computational space. The methods used to handle these processes are not entirely relevant to a project of this scope, as the proposed state space will be orders of magnitude less complicated. Despite this, details within the work pertaining to supervised learning provide excellent example practices which can be transferred to the development of this project.

Project Plan (Gantt Chart):



	Task Name	Start Date	End Date
1	Data Structures	30/10/17	05/11/17
2	Rule/Goal Repo	06/11/17	19/11/17
3	GamePlan	20/11/17	26/11/17
4	Impliment Player	27/11/17	24/12/17
5	Impliment Player	08/01/18	21/01/18
6	Test Player	15/01/18	21/01/18
7	Impliment Architect	22/01/18	11/03/18
8	Teach Architect	05/03/18	01/04/18
9	Produce GamePlans	19/03/18	15/04/18
10	Test GamePlans	16/04/18	26/04/18

Project Plan (Task Breakdown):

The following list describes the justification for the proposed project plan. It is ordered chronologically, and according to the task ID from the above Gantt chart. The items present relate directly to the Objectives section detailed previously in this work.

1. **Data Structures (30/10/17 – 05/11/17)**
   Objective 1: *Instantiate basic data structures and code framework.*

   Program modular python files to handle objects such as playing cards and decks of cards. Set up a Git repository to handle file management.

2. **Rule/Goal Repo (06/11/17 – 19/11/17)**
   Objective 2: *Create a repository of all possible card game 'Rules' and 'Goals' available within the system.*

   Two new data structures are defined, constructed using the basic structures instantiated in the previous objective. The effect rules and goals themselves will be sourced from various real card games.

3. **GamePlan (20/11/17 – 26/11/17)**
   Objective 3: *Instantiate the GamePlan object type, primarily composed of a subset of rules and goals.*

   Constructed using the data structures defined in the previous objective. This object type will be the eventual output of the system from Architect.

4. **&5. Impliment Player (27-11/17 – 24/12/17,** [Christmas], **08/01/18 – 21/01/18)**
   Objective 4: *Impliment Player.*

   A large portion of the allocated time will be devoted to the research, development and debugging of the first AI agent, Player. Near the end of this phase, it will run concurrently with the next objective to help finalise production of the agent.

6. **Test Player (15/01/18 – 21/01/18)**
   Objective 5: *Test Player.*

   A testing plan will be designed concurrently with the development of Player to ensure it is fully applicable to the final design of the agent. This plan will be carried out in the following week, ultimately assessing whether Player will be capable of teaching Architect in the coming objectives.

7. **Impliment Architect (22/01/18 – 11/03/18)**
   Objective 6: *Impliment Architect.*

   The other significant portion of allocated time, this time in regard to the second AI agent, Architect. In this case, implimentation will run concurrently with the next objective as it is a necissary step to completing Architect's development.

8. **Teach Architect (05/03/18 – 01/04/18)**
   Objective 7: *Use Player to teach Architect.*

   Once Architect's main development is completed, the process of developing a 'brain' can begin, allowing Architect to begin assigning weights to combinations of rule and goal objects within GamePlans.

9. **Produce GamePlans (19/03/18 – 15/04/18)**
   Objective 8: *Have Architect produce new gameplans.*

   After sufficient training, Architect's output will be well defined and the agent will be able to generate example GamePlans. Any final alterations to this endpoint will occur in this timeframe, ensuring it is at a standard able to perform testing in the next objective.

10. **Test GamePlans (16/04/18 – 26/04/18)**
    Objective 9: *Test Integrity of Architect's GamePlans.*

    Concluding the project, the GamePlans generated by Architect will be tested by various means to ensure the system is working as intended. Any issues arising at this point should be related to obscure or erroneous rule/goal combinations, meaning they will be rare if present at all.

Risk Assessment:

Risk	Likelihood	Impact	Mitigation
Player lacks required skill to play through GamePlans	Moderate	High	Devote more time to Player's implimentation, upgrading aspects from pseudo-random to deterministic.
Time complexity of the system becomes unmanagable whilst using Player to teach Architect	Low	Moderate	Improve integration between the two agents. Attempt to refactor any CPU-intensive code.
Player's cannot produce suitable heuristics to teach Architect	Low	Moderate	Modifications to Player would allow it to act as a basic classifier despite this. Complex behaviors will be much harder to produce.
Complications with Rule and Goal objects resulting in unsuitable system outputs.	Moderate	Low	Design for these objects must be reworked to better fit the specifications of both AI agents, rather than primarily focusing on Player's interaction with them.
Architect unable to generate valid GamePlans after being trained	Low	High	A large volume of debugging and system checking will need to occur as the fault could lie almost anywhere within the system pipeline.

References

Association for Computational Creativity, 2016. *Computational Creativity.* [Online]
Available at: http://computationalcreativity.net/home/
[Accessed 22 19 2017].

Bansal, T., 2017. *Emergent Complexity via Multi-Agent Competition. San Franciso.*

DeepMind, 2017. *AlphaGo.* [Online]
Available at: https://deepmind.com/research/alphago/
[Accessed 21 10 2017].

Littman, M., 1994. Markov games as a framework for multi-agent reinforcement learning. *ICML,* 11(Proceedings of the Eleventh International Conference on International Conference on Machine Learning), pp. 157-163.

OpenAI, 2017. *Dota 2.* [Online]
Available at: https://blog.openai.com/dota-2/
[Accessed 21 10 2017].

Richard S. Sutton, A. G. B., 1998. *Reinforcement Learning: An Introduction.* London: MIT Press.

Tan, M., 1993. *Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents.* Waltham: GTE.

Wikipedia, 2017. *Richard S. Sutton.* [Online]
Available at: https://en.wikipedia.org/wiki/Richard_S._Sutton
[Accessed 22 10 2017].