Aprendizaje automatizado de Videojuegos

Damián Ariel Marotte

Resumen y motivacion

El presente trabajo pretende mostrar ejemplos de los modelos simbolico y conexionista de inteligencia artificial, asi como tambien exponer la diferencia entre *narrow AI* y *full AI*. Para ello se analizan dos aplicaciones de aprendizaje automatizado de videjuegos: una de ellas es una aplicacion diseñada especificacmente para aprender a jugar un juego en particular, mientras que la otra pretende hacer lo mismo pero de forma general.

I. A. Simbolica

Descripción

learnfun y *playfun* son dos programas desarrollados por el doctor en ciencias de la computación Tom Murphy VII cuyo objetivo es aprender a jugar cualquier videojuego que se le presente. El proyecto esta dividido en dos programas:

- learnfun: Programa destinado a observar un juego e intentar determinar la nocion de *ganar*.
- playfun: Programa que busca una secuencia de entradas para lograr el objetivo del juego.

Para facilitar los costos computacionales, se decidio realizar el trabajo sobre emulacion de una maquina NES. La NES es una plataforma con un procesador de 8 bits a una velocidad de 1.79mhz y 2mb de RAM. Como el poder de computo de las computadoras actuales es mucho mayor, la convierte en una plataforma ideal para investigar estas cuestiones.

learnfun

Uno de los aspectos mas complicados del proyecto es definir una nocion generalizada de *ganar* en un videojuego pues, no es lo mismo progresar en una partida de *Tetris* que en una partida de *Pacman*.

El enfoque utilizado por el autor consiste en observar como evoluciona la memoria a lo largo de una partida. Para ganar en un videojuego, aquellas direcciones que guardan informacion sobre la cantidad de vidas, puntaje, nivel, posicion, etc; deberan incrementarse con el tiempo. En consecuencia, la nocion general de *ganar* es maximizar dichas direcciones de memoria.

playfun

Una vez que *learnfun* a logrado determinar que direcciones de memoria deben maximizarse para progresar en un videojuego, *playfun* genera secuencias de entradas y observa dichas direcciones de memoria; si dichas direcciones aumentan entonces la secuencia se conserva y en caso contrario se prueba otra combinacion.

Una busqueda al azar no solo no es ineficiente, sino que tampoco es util pues la mayoria de las entradas generadas al azar ni siquiera son entradas validas. En cambio se utiliza un modelo probabilistico que determina que entradas son mas probables de funcionar, basandose en las entradas provistas a learnfun.

Conclusiones

Personalmente considero que los resultados obtenidos por el programa son asombrosos. No solo sorprende que la nocion general de *ganar* en un videojuego funcione, sino que tambien es admirable como el programa descubre formas de jugar que nunca se me hubieran ocurrido. Esto me lleva a percibir el comportamiento del programa, como una forma de pensamiento.

Desafortunadamente el caracter genérico del algoritmo, lo hace altamente ineficiente.

I. A. Conexionista

Descripción

crAIg es un programa desarrollado por Niko Savas y Joe Crozier en el marco de la YHack 2014, un encuentro anual de programadores organizado por la universidad de Yale. Implementa un algoritmo genetico de auto-aprendizaje que juega a Super Mario Bros. para la NES.

Desarrollo

Al contrario de los programas descritos anteriormente, crAIg esta basado en NEAT: un algoritmo generico para generar redes neuronales evolutivas. Fue desarrollado en lenguaje Lua, en un plazo de 48 horas.

El programa comienza sin ningun conocimiento del juego ni como jugarlo y apoyandose en una funcion que calcula la distancia que recorrio el personaje, el algoritmo es capaz de aprender a moverse y esquivar obstaculos para superar un nivel.

Conclusiones

crAIg tiene una ventaja evidente sobre learnfun y playfun: es considerablemente mas rapido, cuestion que resulta razonable teniendo en cuenta que solo se focaliza en un solo juego. Aunque NEAT es un algoritmo generico, para que pueda aprender otro juego es necesario adaptarlo. Por el otro lado si bien playfun y learnfun son extremadamente ineficientes, su caracter general los transforman en algoritmos de gran interes.

En ambos casos el problema de la eficiencia parece ser un factor fundamental: los tiempos en los que ambas aplicaciones aprenden fueron, muy superiores a lo que personalmente esperaba. Puesto que los dos programas demostraron cumplir con sus objetivos, considero que la investigacion debe dirigirse hacia tecnicas de optimizacion.

Referencias

- Tom Murphy. *The First Level of Super Mario Bros. is Easy with Lexicographic Orderings and Time Travel... after that it gets a little tricky.* 1 de Abril de 2013.
- *learnfun & playfun website.*
- *learnfun & playfun source code.*
- *crAIg website.*
- Kenneth O. Stanley & Risto Miikkulainen. *Evolving Neural Networks throughAugmenting Topologies.*
- *crAIg source code.*