Trabajo práctico 1:
"Algo2landia"

Normativa

Límite de entrega: Domingo 19 de Septiembre, 23:59hs. Subir el pdf al repo grupal
Normas de entrega: Ver "Información sobre la cursada" en el sitio Web de la materia. (`campus.exactas.uba.ar`)
Versión: 1.0 del 23 de agosto de 2021 (ver CHANGELOG-tp1.md)

Enunciado

Este trabajo práctico plantea modelar las reglas de un mundo simulado para el entrenamiento de agentes inteligentes llamado *algo2landia*. El mismo está basado en los mundos de entrenamiento desarrollados por OpenAI [1] o XLand de DeepMind [2] .

El enunciado se divide en dos partes que conforman una solución incremental del resultado completo. A fines didácticos, proponemos resolver las partes en órden. Además, pondremos una fecha de entrega temprana de la primera parte de forma que puedan tener una devolución de la misma antes del primer parcial. Esto es para que la instancia les sirva de validación del aprendizaje y consulta de los contenidos del mismo.

Parte 1

Algo2landia se constituye como una grilla de casilleros cuadrados de dimensiones finitas. Cada casillero puede tener tres tipos de altura: piso, rampa o elevación. Un casillero puede ser una rampa solo si está adyacente (sin contar diagonales) a por lo menos un casillero piso y un casillero elevación.

Sobre este mundo se produciran simulaciones en donde actuará el agente — prontamente —inteligente. Una simulación comienza con el agente en algún casillero. El mismo puede realizar una única acción: moverse en alguna de las cuatro direcciones cardinales (arriba, abajo, izquierda, derecha). El agente no podrá moverse por fuera de los límites del mapa y tampoco puede escalar a una elevación si no es mediante una rampa. Para bajar no tendrá restricciones, pudiendo caer desde una elevación al piso o bajar mediante una rampa.

Además, nos interesa conocer cuantos movimientos intentó realizar el agente durante la simulación. Esto significa que todos los movimientos, incluso aquellos no realizados por chocarse con una pared, se contabilizan. La única excepción son los intentos de movimientos hacia el exterior del mundo.

Parte 2

Como parte del mundo también se definirán casilleros especiales que estarán pintados de color. Estos casilleros pueden ser de cualquier altura. No hay límite a la cantidad de colores distintos que puede haber, por lo que se los identificará con un nombre. En la simulación también habrá objetos, que ocuparán el espacio de una casilla. Los mismos también estarán pintados de un color. Los colores de los objetos también se identificarán con un nombre y pueden o no repetirse con los colores de casilleros. No habrá dos objetos o dos casilleros del mismo color.

Si al moverse el agente, se encuentra con un objeto, el mismo se empujará un casillero en la dirección del movimiento. Para los objetos aplican las mismas reglas que para el agente: no pueden subir a elevaciones si no es mediante una rampa y no pueden caerse del mapa. Si el movimiento del agente llevase a estas situaciones, no podrá hacerse, aunque se contabilizará como intento de movimiento. Finalmente, un agente no puede empujar dos objetos a la vez (dos objetos en línea en la dirección de movimiento).

Finalmente, el agente tendrá objetivos a realizar. Los mismos consisten en empujar el objeto de un color o hasta un casillero de color c. Los objetivos pueden agregarse en cualquier momento durante la simulación. En cada momento, el agente tendrá un conjunto de objetivos disponibles para completar. Los mismos se completan el instante que el objeto objetivo alcanza la casilla objetivo, dejando de estar disponible para completarse hasta que se lo agregue nuevamente. Al completar un objetivo, el agente estará libre para tratar de completar el resto de los objetivos disponibles. De no tener objetivos disponibles, es libre de moverse a la espera de nuevas tareas. Nos interesa conocer cuántos objetivos tiene realizados el agente en una simulación, ya que la proporción de objetivos realizados en cantidad de pasos será la forma de medir el rendimiento del agente.

[1] `https://openai.com/blog/emergent-tool-use/`
[2] `https://bdtechtalks.com/2021/08/02/deepmind-xland-deep-reinforcement-learning/`

Entrega

Fechas de entrega

La **primera parte** del TP se entrega el Domingo 5 de Septiembre hasta la medianoche. La propuesta es que completen la primera parte del enunciado para esa fecha, para revisarla con su corrector/a asignado/a el Miércoles 8 de Septiembre. En caso de no lograr finalizar la primera parte, entregar el avance logrado con un commit en su repositorio grupal. Esto sirve, además de para tener una devolución de su corrector/a, para acostumbrarse al mecanismo de entregas.

El TP **completo** se debe entregar antes del Domingo 19 de Septiembre a las 23:59. Su devolución será hasta el Miércoles 29 de Septiembre, en horario de consulta, inclusive. Para la devolución deberán estar presente el equipo completo. En caso de tener dificultades con presentarse, comunicarse con su corrector/a.

Formato de entrega

Para la entrega deben hacer `commit` y `push` de un único documento digital en formato `pdf` en el repositorio **grupal** en el directorio `tpg1/`. El documento debe incluir la especificación completa del enunciado presentado usando el lenguaje de especificación con TADs de la materia. Se recomienda el uso de los paquetes de LaTeX de la cátedra para lograr una mejor visualización del informe.

Rubricas

Agregamos a continuación rúbricas que exponen qué se espera de la entrega. Las mismas presentan una serie criterios con los que se evaluarán las producciones entregadas. En términos generales, se considera que entregas con soluciones que solo logren los criterios parcialmente deberán ser reentregados con correcciones en estos aspectos en particular.

Por ser criterios generales, pueden no cubrir todos los detalles relacionados con este enunciado. No obstante buscamos que sean lo más completas posibles. Esperamos que las mismas les sirvan de orientación para la resolución del TP.

	Logra Totalmente	Logra	Logra Parcialmente	No Logra
Abstraccion	Los TADs capturan todos los elementos relevantes del enunciado, y NO capturan elementos irrelevantes.	Los TAD capturan todos los elementos relevantes del enunciado, pero capturan elementos irelevantes.	Los TAD NO capturan todos los elementos relevantes del enunciado.	Los TAD NO capturan todos los elementos relevantes del enunciado y modelan cosas no relacionadas con él.
Abstraccion funcional (o modularización)	Los TADs tienen una responsabilidad adecuada (ni mucha ni poca) y no hay funciones muy extensas. Cuando es adecuado, hay más de un TAD de forma de separar el problema.	Los TADs tienen una responsabilidad adecuada pero hay funciones muy extensas que podrían separarse en funciones auxiliares.	Hay TAD(s) que acumulan más responsabilidad de la necesaria, pero la formulación es correcta y no dificulta la comprensión del modelado.	Hacen todo en un TAD y no es correcto o no se puede entiende sin leer toda la axiomatizacion (es ilegible, no es una especificacion para un humano).
El Comportamiento Automático (CA) es adecuado	Para todos los CA NO es necesario aplicar un generador para que sea efectivo.	Falta algun CA pero es justificable con una interpretacion distinta del enunciado sin simplificar enormemente el mismo.	Falta algún CA que simplifica el enunciado.	No hay CA (y debería haberlo).
Sobreespecificacion	No hay especificacion de aspectos no definidos (por ejemplo, poner una lista donde va un conjunto) ni restricciones que sobresimplifican en problema (por ejemplo, asumir cierto dominio en los parametros).	Hay restricciones innesesarias sobre el dominio pero no sobresimplifica el enunciado (por ejemplo Ponen Nat pero podria ser Int).	Sobreespecifica (por ejemplo: poner lista cuando cuando no hay orden en los el los elementos, tengo una lista de jugadores).	Se sobresimplifica el enunciado.
Declaratividad	Los nombres de las funciones principales (generadores y observadores) ayudan a entender el dominio modelado.	Los nombres de las funciones principales ayudan a entender el modelado pero los axiomas no se condicen con el nombre (ejemplo: existeFantasma() devuelve una lista en lugar de un booleano).	Se abrevia o nombran los metodos de manera que solo se entiende leyendo toda la axiomatizacion.	Se abrevian los nombres y ademas la axiomatizacion no utiliza ninguna funcion auxiliar que ayude a entender por donde van (es ilegible, no es una especificacion para un humano).
Observadores Minimales	Los observadores son mininales y representan todas las instancias posibles validas.	Los observadores NO son mininales, representan todas las instancias posibles validas, pero NO rompe congruencia.	Los observadores NO son mininales, NO representan todas las instancias posibles validas, pero NO rompe congruencia.	Los observadores NO son mininales, NO representan todas las instancias posibles validas, y rompe congruencia.
Correctitud en general	No mezclan generadores entre TAD ni dejan combinaciones de obs/gen sin resolver. Los axiomas terminan y tipan bien.	Algún comportamiento no está perfectamente reflejado en la axiomatización (bugs), hay errores de tipo leves o brazos de condicionales sin resolver. Ninguno de estos errores afectan a la comprensión de la espeficiación.	Hay funciones sin axiomatizar. Hay errores en la axiomatización que dificultan la comprensión de la solución.	Hay funciones sin axiomatizar o errores en los mismos que dejan partes del modelado sin resolver.
Restricciones en funciones	Se aclaran restricciones en funciones que no son necesarias para que la función pueda resolverse pero que acotan correctamente el dominio del problema.	Se detectan las funciones parciales y se restringe su dominio correctamente.	Hay funciones que deberian ser parciales pero no lo son pero su comportamiento extendido no genera comportamientos inválidos.	Hay funciones que deberian ser parciales pero no lo son, permitiendo comportamientos que no pueden resolverse o violan el enunciado.