Jia Long Ji Qiu
Victor Teixidó López
Jiabo Wang
You Wu

PROBLEMA 2 SID

Decisiones de diseño

El agente Player para esta práctica, lo hemos diseñado como un agente racional con la capacidad de decidir si coopera o no con otros agentes dentro del contexto del problema del dilema del prisionero.

A la hora de crear el agente, este recibe un total de 5 parámetros. Los 4 primeros, definen la tabla de penalizaciones que representan todas las jugadas que pueden darse entre dos agentes (CC, CD, DC y DD). El 5 parámetro, es un entero que indicará qué estrategia seguirá el agente para tomar las decisiones durante las partidas. Por un lado, cuando toma por valor 0, dicho agente sigue la estrategia simple descrita en el enunciado, por otro lado, cuando el valor es 1 se utiliza una estrategia implementada basada en el diseño de *Tit for tat*. Adicionalmente, a la hora de diseñar los agentes, cabe destacar que hemos supuesto que las tablas de penalizaciones de todos ellos son idénticas.

En referencia al estado interno del agente, hemos decidido crear una única *capability*, que engloba todas las creencias que definen el sistema en el que se encuentra el agente. Dentro de su base de creencias, o *BeliefBase*, hemos creado un conjunto *beliefs* para poder encapsular los siguientes conceptos:

- La tabla de penalizaciones
- El histórico
- La penalización total acumulada
- Los agentes con los que aún no se ha iniciado ninguna partida
- Los agentes con los que ya se ha iniciado una partida

Hay otros aspectos para los cuales no hemos creado *beliefs* para encapsular los elementos, como por ejemplo el nombre del agente, el DFAgentDescription, el AID o el DFService, ya que son elementos relacionados con la propia plataforma FIPA.

Respecto a la inicialización de partidas entre los distintos agentes de la plataforma hemos acabado implementando un diseño que no fue el inicialmente pensado. Al principio, según lo especificado en el enunciado, decidimos que los agentes debían tener partidas únicas entre ellos. Para ello, antes de proponer una jugada a un agente rival, un agente debía comprobar que en la cola de mensajes no hubiera una propuesta pendiente del rival, y en el caso de haberla, se descartaba la propuesta. Sin embargo, había casos en los que los mensajes no llegaban a tiempo, y ambos agentes se proponían igualmente una jugada, dando resultado a dos conversaciones. Finalmente, optamos por una implementación en la que entre dos agentes existen dos conversaciones simultáneas, cada una empezada por uno de los dos y, en los históricos, las jugadas están guardadas en función del identificador de la conversación entre agentes, definidos como el AID del agente que propone la jugada concatenado con “ vs. “ y el AID del agente que recibe la jugada. Esto permite que las penalizaciones y los históricos no se vean alterados en absoluto.

Para los 4 objetivos especificados en el enunciado, hemos decidido implementar 4 plan bodies, uno para cada uno de estos:

- RegistrarAgentePlanBody
- BuscarAgentesPlanBody
- ProponerJugadaPlanBody
- EsperarJugadaPlanBody

RegistrarAgentePlanBody se asocia con un *PredicateGoal* llamado *RegistrarAgente*. Este plan es uno finito, que consiste en el propósito de registrar al agente en la plataforma a través del *DirectoryFacilitator*, dicho agente, se registrará con serviceType == '*player'*. Una vez registrado correctamente, el predicado *RegistrarAgente* pasa a ser true, para indicar que se ha cumplido el objetivo.

En *BuscarAgentesPlanBody*, el objetivo reside en encontrar a agentes disponibles en la plataforma con los que poder empezar partidas nuevas. Será necesario que tengan el tipo servicio como '*player'*. Aquellos agentes que ya se encuentren en el belief del conjunto de agentes ya jugando se ignoran, y el resto se añade al belief del conjunto de agentes pendientes. A diferencia del anterior, este objetivo no terminará nunca, ya que siempre pueden aparecer nuevos agentes con los que poder empezar una partida nueva.

El plan *ProponerJugadaPlanBody*, busca empezar partidas con agentes con los que aún no hayan empezado una partida, recordemos que para cada par de agentes habrán dos conversaciones. Como parece evidente, este plan tampoco tendrá final ya que, cuando aparezca un agente nuevo en la plataforma, tendremos por objetivo empezar una nueva partida. De este plan deriva además el basado en *Tit For Tat*.

El cuarto plan, *EsperarJugadaPlanBody*, tiene como finalidad la continuación de partidas ya empezadas y, por tanto, también será un plan que no tendrá final. Siempre que se reciba una respuesta, a través de este plan el agente decidirá qué responder y enviará la jugada. De este plan deriva además el basado en *Tit For Tat*.

Es importante mencionar que existe una jerarquía entre los objetivos. Los objetivos de buscar agentes, proponer jugadas y esperar jugadas se pueden ejecutar de manera paralela, pero no sin antes haber registrado el agente. Por lo tanto, estos tres objetivos se pueden agrupar en un ParallelGoal, y dicho ParallelGoal se debe agrupar con el objetivo de registrar el agente, formando un SequentialGoal (con el objetivo de registrar el agente en la primera posición).

Jia Long Ji Qiu
Victor Teixidó López
Jiabo Wang
You Wu

Extra

Para el apartado extra hemos implementado una estrategia *Tit For Tat*. Esta estrategia consiste en cooperar en la primera iteración de una partida, mientras que en el resto se copia la jugada que ha elegido el agente rival.

En relación a cómo se ha implementado dicha estrategia, con tal de tener un código más limpio, hemos creado dos plan bodies más que tienen como base los plan bodies *ProponerJugadaPlanBody* y *EsperarJugadaPlanBody*, llamados *ProponerJugadaTFTPlanBody* y *EsperarJugadaTFTPlanBody*. La diferencia fundamental está en cómo se deciden las jugadas, y se hace según la definición de la estrategia TFT:

- La jugada propuesta siempre es cooperar
- La respuesta a una jugada será cooperar si es la primera iteración, y en caso contrario será la misma jugada recibida

Reparto del trabajo

El cuerpo y estructura del agente la hemos realizado entre los 4 miembros del equipo ya que fue lo primero que hicimos y donde más decisiones se tenían que tomar inicialmente. Respecto a los diferentes planes, si hemos optado por una división del trabajo en parejas.

Planes *RegistrarAgentePlanBody* y *ProponerJugadaPlanBody*:

- Jia Long Ji Qiu
- Jiabo Wang

Planes *BuscarAgentesPlanBody* y *EsperarJugadaPlanBody*:

- Victor Teixidó López
- You Wu

Planes *ProponerJugadaTFTPlanBody* y *EsperarJugadaTFTPlanBody*:

- Jia Long Ji Qiu
- Victor Teixidó López