DILEMA DEL PRISIONERO CON JUGADORES EVOLUTIVOS

- Carlos Andrés Delgado S. 0831085
- Víctor Alberto Romero G.. 0632725

1. Descripción del problema.

El problema a solucionar consiste en implementar el dilema del prisionero utilizando jugadores evolutivos, es decir cuyo comportamiento evolucione de acuerdo a las interacciones con otros jugadores.

2. Solución propuesta.

2.1 Representación de datos.

Jugadas:

Las jugadas se representan asi:

0 para Traicionar
1 para cooperar

Maquina de estados:
Cada jugador cuenta con una máquina de estados cuya entrada es la jugada del oponente y su salida es su jugada. De acuerdo a la entrada se pasa a un próximo estado.

Ejemplo:

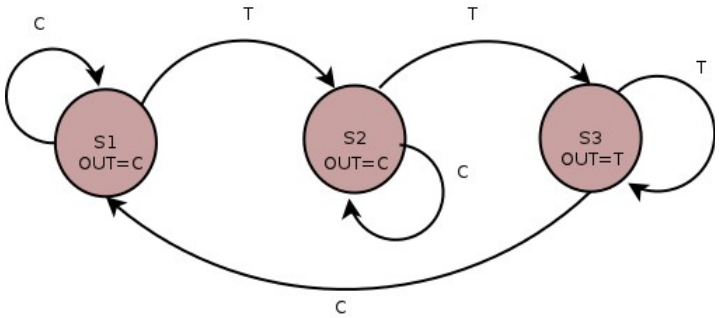

Figura 1. Maquina de estados de un jugador.

Se puede observar que inicialmente el jugador inicia en SI, y su salida es cooperar, pero si el oponente juega Traicionar pasa al estado S2.

Esta maquina de estados se representa de la siguiente manera:

[EstadoPresente Salida ProximoEstadoSiCooperan ProximoEstadoSiTraicionan]

En la aplicación se representa como un arreglo bidimensional de la siguiente manera:

[
[1 1 1 2]
[2 1 2 3]
[3 0 1 3]
]

Cruce

El cruce se realiza seleccionado dos jugadores aleatoriamiente, se establece que su numero de estados es un número entre 2 y la suma del numero de estados de sus padres, estos se combinan seleccionando primero los del jugador de menor numero de estados y posteriormente del mayor del número de estados, si un estado apunta a un estado próximo mayor que el numero de estados del hijo este se cambiará por el número de estados, es decir apuntaran al ultimo estado.

Mutación.

Se seleccionan el 10% de los jugadores al azar, después se mutan un número aleatorio de estados del jugador, los valores que pueden cambiar de un estado a mutar son: salida, próximo estado si traicionan y próximo estado si cooperan.

Limpieza y reorganización de estados.

Para la limpieza de estados se realiza:

1. **Búsqueda de estados conexos:** Es decir, aquellos estados que se pueden ver con cualquier combinación de jugadas del oponente, el resto de estados es descartado ya que no son alcanzables.
2. **Reorganización de maquinas de estado:** Los índices de Estado presente y próximo estados son reorganizados una vez son eliminados los estados no alcanzables, para que la maquina de estados sea consistente.

Función de aptitud.

La función de aptitud es seleccionar el 60% de los jugadores que obtienen mayor ganancia al jugar un número aleatorio de partidas todos contra todos.

Existe una penalización del número de estados que consiste en restar el 10% de la diferencia entre el numero de estados y el promedio de estados de todos los jugadores de la ganancia total del jugador.

Implementación.

La solución del problema se implementa utilizando C++ con Qt.

Clases.

Estado: Esta clase genera un estado aleatoriamente de la forma [X Y Z W]
X -> Es generado secuencialmente entre 1 y numero de estados.
Y -> Es 0 o 1.
Z -> Es generado aleatoriamente entre 1 y numero de estados.
W -> Es generado aleatoriamente entre 1 y numero de estados.

Maquina de estados: Es un arreglo bidimensional de estados, esta clase provee las funciones necesarias para crear, editar y ver los estados.

Matriz de pagos: Contiene los valores de la matriz de pagos, retorna la ganancia de un jugador de acuerdo a su jugada y la del oponente.

Prisionero: Provee las funciones para realizar el juego.

- Crea población de jugadores.
- Los pone a jugar entre ellos y genera las ganancias totales de cada uno.
- Provee las funciones de cruce
- Provee las funciones de mutación.

Aspectos importantes de la implementación.

Los estado se numeran entre 1 y numero de estados, por esta razón se indexa con “Numero estado presente” -1.

Ejemplo:

```
------------------------------------------------------------------------
----------------------   Población Inicial   ----------------------
------------------------------------------------------------------------

La ganancia promedio es: 11359.7
La ganancia del mejor es: 15844
+---------------------------------+
|        Tabla de Estados         |
+---------------------------------+
| EP     S       PEC     PET      |
|                                 |
| 1      T       1       1        |
+---------------------------------+

------------------------------------------------------------------------
----------------------   Población Final   ----------------------
------------------------------------------------------------------------

La ganancia promedio es: 22936.8
La ganancia del mejor es: 23298
+---------------------------------+
|        Tabla de Estados         |
+---------------------------------+
| EP     S       PEC     PET      |
|                                 |
| 1      C       1       2        |
| 2      T       2       2        |
+---------------------------------+
----------------------   Población Inicial   ----------------------
------------------------------------------------------------------------

La ganancia promedio es: 10969
La ganancia del mejor es: 14816
+---------------------------------+
|        Tabla de Estados         |
+---------------------------------+
| EP     S       PEC     PET      |
|                                 |
| 1      T       3       4        |
| 2      T       1       2        |
| 3      T       4       2        |
| 4      T       5       4        |
| 5      T       5       5        |
+---------------------------------+

------------------------------------------------------------------------
----------------------   Población Final   ----------------------
------------------------------------------------------------------------

La ganancia promedio es: 20923.8
La ganancia del mejor es: 22140
+---------------------------------+
|        Tabla de Estados         |
+---------------------------------+
| EP     S       PEC     PET      |
|                                 |
| 1      T       2       2        |
| 2      C       2       3        |
| 3      C       3       4        |
| 4      T       5       5        |
| 5      T       4       4        |
```

Figura 2. Ejemplos de ejecución.

Como se puede observar en este ejemplo, en el primer caso se obtiene que esta maquina coopera a la primera vez y traiciona si la traicionan una vez; en el segundo caso el jugador tiene una maquina de estados que inicialmente traiciona pero si coopera este cooperará, si siguen cooperando esté cooperará, si lo traicionan cooperará, pero si lo vuelven a traicionar, este traicionará sin importar la jugada del oponente.

Conclusiones.

- Los jugadores evolutivos tienden inicialmente a traicionarse entre sí y los que cooperan inicialmente son descartados de la población, sin embargo debido a los cruces y mutaciones entre jugadores, se encuentra que un buen número de veces los jugadores tienden a maquinas de estado que cooperan o bien que cooperan una vez y traicionan si son traicionadas.

- En algunos casos arroja como resultado inicial que las maquinas de estado se traicionan, esto se debe que en la población inicial un gran porcentaje traicionaban por lo que en las primeras generaciones los que cooperaban fueron extinguidos. Sin embargo si se **aumenta considerablemente el número de generaciones** el sistema tenderá a la cooperación y se observan resultados muy interesantes, como maquinas de estado que perdonan una traición pero no dos, entre otras.

- La función de aptitud y la búsqueda de estados conexos, tienden en todos los casos a que los jugadores tengan maquinas con pocos estados, sin embargo **no es determinante** en la cooperación o traición de los jugadores a medida que evolucionan.

- A pesar de que no en todas las simulaciones se encontró cooperación, se pudo establecer que si se aumentan el número de generaciones, se observa que la evolución **si lleva a la cooperación,** debido a que las **mutaciones** a la larga hacen tender el sistema a la cooperación.