

Planificación en Entornos con incertidumbre

Inteligencia Artificial - Curso 2018/2019
<https://github.com/a8081/ia>

José Manuel González Gutiérrez
Universidad de Sevilla (Sevilla, España)
UVUS: josgongut1 (josgongut1@alum.us.es)

Antonio Martínez Rojas
Universidad de Sevilla (Sevilla, España)
UVUS: antmarroj (antmarroj@alum.us.es)

Agosto 2019

Resumen - El objetivo principal de este trabajo es entender la Planificación en entornos con incertidumbre, con esto se busca ampliar el contenido visto en la asignatura, además, de conseguir que los alumnos hagan un pequeño trabajo de investigación por ellos mismos.

1 INTRODUCCIÓN

Para introducir este trabajo vamos a comenzar explicando que se entiende por planificar o planificación.

“Planificar consiste en encontrar una secuencia de acciones para alcanzar un determinado objetivo cuando se ejecutan a partir de un determinado estado inicial.”[4]

Un proceso de decisión de Markov parcialmente observable (POMDP) es una generalización de un proceso de decisión de Markov (MDP). Un POMDP modela un proceso de decisión del agente en el que se supone que la dinámica del sistema está determinada por un MDP, pero el agente no puede observar directamente el estado subyacente. En cambio, debe mantener una distribución de probabilidad sobre el conjunto de estados posibles, basada en un conjunto de observaciones y probabilidades de observación, y el MDP subyacente.[3]

Con esto, el objetivo principal es solucionar 4 problemas, dos de estos establecidos por el profesor y otros dos elegidos por nosotros. Los problemas propuestos son los siguientes:

- **Problema del Tigre:** Existen dos puertas cerradas, existe un tigre tras una de ellas, en la otra la libertad. Hay que decidir que puerta tomar.[4]
- **Problema del Laser Tag:** Existen dos robot, uno intenta atrapar a otro moviéndose por un tablero.[4]

Estos problemas deben ser resueltos mediante dos algoritmos:

- POMCP
- PBVI

También se deberán hacer distintos modos de ejecución donde se encuentran los siguientes:

- **Simulación Interactiva:** Se muestra cada acción tomada por el algoritmo y el estado en el que se encuentra.
- **Simulación Silenciosa:** Se muestra el número de pasos y recompensa acumulada cuando se llega a una condición de parada.
- **Benchmark:** Se realizan 30 simulaciones silenciosas, mostrando al final valores medios, desviaciones típicas y gráficas.

2 PRELIMINARES

2.1 Métodos empleados

- **PBVI:** El algoritmo de iteración de valores para la planificación en procesos de decisión de Markov parcialmente observables (POMDPs) fue introducido en la década de 1970[Sondik, 1971]. Desde su introducción numerosos autores lo han refinado[Cassandra et al., 1997; Kaelbling y otros, 1998; Zhang y Zhang, 2001] para que pueda resolver problemas más difíciles. Pero a día de hoy, los algoritmos de iteración de valores de POMDP no son capaces de escalar a problemas del mundo real. [6] Por este mismo motivo, algunos de nuestro problemas han tenido que ser bloqueados para este algoritmo ya que no se conseguían resolver.
- **POMCP:** POMCP combina las actualizaciones del estado de las creencias de Monte-Carlo con PO-UCT, y comparte las mismas simulaciones para ambos procedimientos de Monte-Carlo. Cada simulación comienza desde un estado de inicio que es muestreado desde el estado de creencia. Las simulaciones se realizan utilizando el algoritmo UCT parcialmente observable. Por cada historia encontrada durante la simulación, el estado de

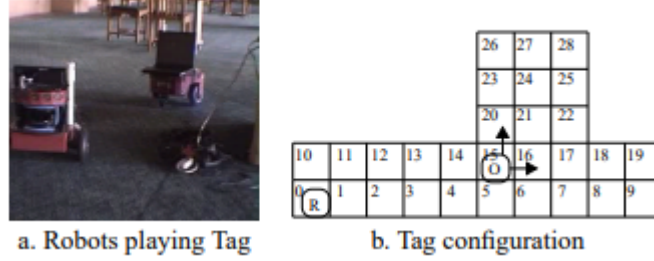


Figure 1: Laser Tag (870 estados, 5 acciones, 30 observaciones)[6]

creencia se actualiza, para incluir el estado de simulación. Una vez finalizada la búsqueda, el agente selecciona la acción con el mayor valor de creencia y recibe una observación real del mundo. En este punto, el nodo:

$$T(h_t a_t o_t)$$

Se convierte en la raíz del nuevo árbol de búsqueda, y el estado de creencia

$$B(h_t a_t o_t)$$

determina el nuevo estado de creencia del agente. El resto del árbol se poda, ya que todas las demás historias son ahora imposibles. El algoritmo completo de POMCP se describe en el pseudocódigo y la figura mostrada a continuación.[9][1]

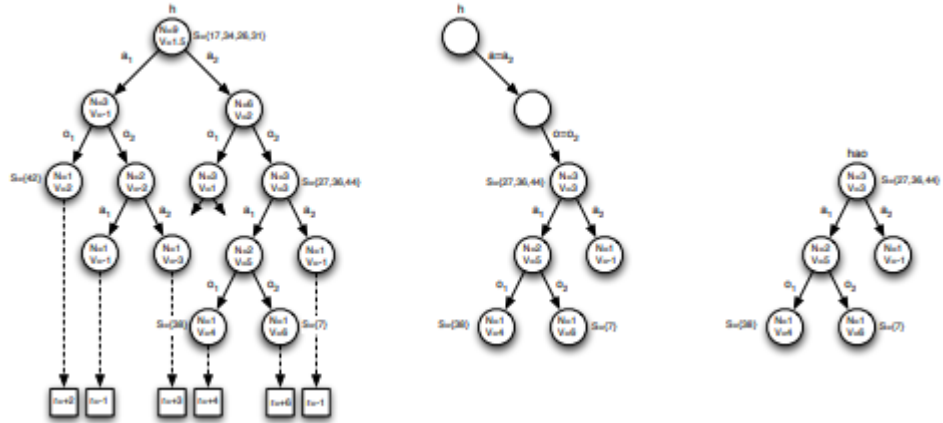


Figure 2: Una ilustración de POMCP en un entorno con 2 acciones, 2 observaciones, 50 estados y sin recompensas intermedias.[9]

Algorithm 1 Partially Observable Monte-Carlo Planning

<pre>procedure SEARCH(h) repeat if $h = \text{empty}$ then $s \sim \mathcal{I}$ else $s \sim B(h)$ end if SIMULATE($s, h, 0$) until TIMEOUT() return $\underset{b}{\operatorname{argmax}} V(hb)$ end procedure procedure ROLLOUT(s, h, depth) if $\gamma^{\text{depth}} < \epsilon$ then return 0 end if $a \sim \pi_{\text{rollout}}(h, \cdot)$ $(s', o, r) \sim \mathcal{G}(s, a)$ return $r + \gamma \cdot \text{ROLLOUT}(s', hao, \text{depth}+1)$ end procedure</pre>	<pre>procedure SIMULATE(s, h, depth) if $\gamma^{\text{depth}} < \epsilon$ then return 0 end if if $h \notin T$ then for all $a \in \mathcal{A}$ do $T(ha) \leftarrow (N_{\text{init}}(ha), V_{\text{init}}(ha), \emptyset)$ end for return ROLLOUT(s, h, depth) end if $a \leftarrow \underset{b}{\operatorname{argmax}} V(hb) + c\sqrt{\frac{\log N(h)}{N(hb)}}$ $(s', o, r) \sim \mathcal{G}(s, a)$ $R \leftarrow r + \gamma \cdot \text{SIMULATE}(s', hao, \text{depth} + 1)$ $B(h) \leftarrow B(h) \cup \{s\}$ $N(h) \leftarrow N(h) + 1$ $N(ha) \leftarrow N(ha) + 1$ $V(ha) \leftarrow V(ha) + \frac{R - V(ha)}{N(ha)}$ return R end procedure</pre> <hr/>
--	--

Figure 3: Planificación Parcialmente Observable en Monte-Carlo

3 METODOLOGÍA

En este apartado hablaremos sobre la metodología de trabajo, esta se divide en dos partes, investigación teórica e investigación enfocada a la implementación. Cabe destacar que esta primera se llevo a cabo en más profundidad por una desviación inicial en el enfoque del trabajo.

3.1 INVESTIGACIÓN TEÓRICA

En esta primera toma de contacto y debido a que no teníamos conocimientos sobre POMDP, comenzamos una investigación puramente teórica de dicho método. Lo que implico que las primeras semanas de trabajo se centraran en entender como funcionaba y cual era la idea de este enfoque. Comenzamos las primeras implementaciones de código inspirándonos en la librería[8] ya que pensamos que ese era el fin en lugar de hacer un punto de entrada para la libreria. Esta librería no seria usada finalmente.

Algunos recursos de los que usamos fue el material encontrado en la web del departamento (Diapositivas que en este momento no consigo encontrar para hacer referencia) y algunos vídeos de JL Iglesias Fera.[5]

3.2 INVESTIGACIÓN ENFOCADA A LA IMPLEMENTACIÓN

Tras comenzar un nuevo enfoque comenzamos a elegir librerías. Gracias al punto de investigación anterior tuvimos claro que la librería que usaríamos sería PyPOMDP[7], ya que era la que más nos facilitaba el trabajo tanto a la hora de desarrollar, como de introducir los problemas. Debido a que con esta librería solo era necesario el archivo de especificación (*.POMDP) y ella se encargaría de cargar la información, de esta manera solo era necesario desarrollar los que serían los puntos de entrada y las condiciones de parada específicas para cada problema.

Tras elegir la librería, comenzamos la tarea de búsqueda de problemas e implementación de la interfaz mediante consola. Esta puede ser ejecutada mediante el siguiente comando:

```
$ python ./main.py
```

Algunas capturas de la interfaz mediante terminal [ANEXO I].

En cuanto a los problemas elegidos serían:

- **Problema del Indiana Jones:** Existen tres túneles por los que cae una roca, Indiana Jones tiene que decidir en cual de los tres túneles quedarse para sobrevivir.[ANEXO III]
- **Anuncios web:** Tiene un sitio web que ofrece una gama de productos y desea adaptar los productos ofrecidos o anunciados al tipo de persona que está visitando su sitio. No quiere interferir con su navegación, por lo que las únicas pistas de su personalidad son productos en los que parecen estar interesados.[2]

De algunos de los problemas seleccionados encontramos su archivo de definición, aunque la mayoría hubo que modificarlos (o generarlos desde cero) ya que la librería no admitía ciertas expresiones del lenguaje, como por ejemplo (*) para referirse a cualquier acción.

Tras conseguir que todos los archivos funcionaran, desarrollamos los modos de ejecución, comenzando con la simulación interactiva, simulación silenciosa y por ultimo el modo de ejecución en modo benchmark.

Tras realizar varias pruebas decidimos también realizar una interfaz gráfica que se muestra en el ANEXO II.

4 RESULTADOS

Tras terminar el software comenzamos con las pruebas, ejecutando los distintos problemas con varias configuraciones. Nos encontramos que debido a la ineficiencia de la implementación de PBVI no se podía ejecutar para problemas demasiado grandes, como el Laser TAG, ya mencionado anteriormente.

Sin embargo, PBVI[6] era mucho más rápido en su ejecución que POMCP[9], debido a que su complejidad aumenta exponencialmente conforme aumenta la dimensión del problema. Pero, tiene muy buen rendimiento en problemas como el del tigre.

Realizamos varias pruebas de ejecución, comparando resultados entre distintas ejecuciones, algo bastante sencillo gracias al modo de ejecución benchmark.

A continuación algunas capturas de ejecución y gráficas de benchmark:

```

-----
Problema: IndianaJones
Algoritmo: pbvi
Simulación: benchmark
Nº pasos máximo: 100
Presupuesto: inf
-----
¿Desea continuar? [S]/[N]s
30 | 3 pasos. Recompensa Total = 8.0
29 | 3 pasos. Recompensa Total = 8.0
28 | 3 pasos. Recompensa Total = 8.0
27 | 3 pasos. Recompensa Total = 8.0
26 | 3 pasos. Recompensa Total = 8.0
25 | 6 pasos. Recompensa Total = 5.0
24 | 7 pasos. Recompensa Total = 4.0
23 | 5 pasos. Recompensa Total = 6.0
22 | 3 pasos. Recompensa Total = 8.0
21 | 5 pasos. Recompensa Total = 6.0
20 | 3 pasos. Recompensa Total = 8.0
19 | 3 pasos. Recompensa Total = 8.0
18 | 4 pasos. Recompensa Total = 7.0
17 | 3 pasos. Recompensa Total = 8.0
16 | 3 pasos. Recompensa Total = 8.0
15 | 3 pasos. Recompensa Total = 8.0
14 | 3 pasos. Recompensa Total = 8.0
13 | 3 pasos. Recompensa Total = 8.0
12 | 4 pasos. Recompensa Total = 7.0
11 | 3 pasos. Recompensa Total = 8.0
10 | 3 pasos. Recompensa Total = 8.0
9 | 3 pasos. Recompensa Total = 8.0
8 | 3 pasos. Recompensa Total = 8.0
7 | 3 pasos. Recompensa Total = 8.0
6 | 3 pasos. Recompensa Total = 8.0
5 | 3 pasos. Recompensa Total = 8.0
4 | 3 pasos. Recompensa Total = 8.0
3 | 3 pasos. Recompensa Total = 8.0
2 | 3 pasos. Recompensa Total = 8.0
1 | 3 pasos. Recompensa Total = 8.0
#####
Media de los pasos: 3.433333333333333
Varianza de los pasos: 0.9788888888888895
Desviacion Tipica de los pasos: 0.9893881386437223
Grafica: https://image-charts.com/chart?cht=lc&chd=t:3,3,3,3
#####
Media de la recompensa: 7.566666666666666
Varianza de la recompensa: 0.9788888888888895
Desviacion Tipica de la recompensa: 0.9893881386437223
Grafica: https://image-charts.com/chart?cht=lc&chd=t:8.0,8.0
#####

```

Figure 4: Ejecución del problema de Indiana Jones en modo Benchmark

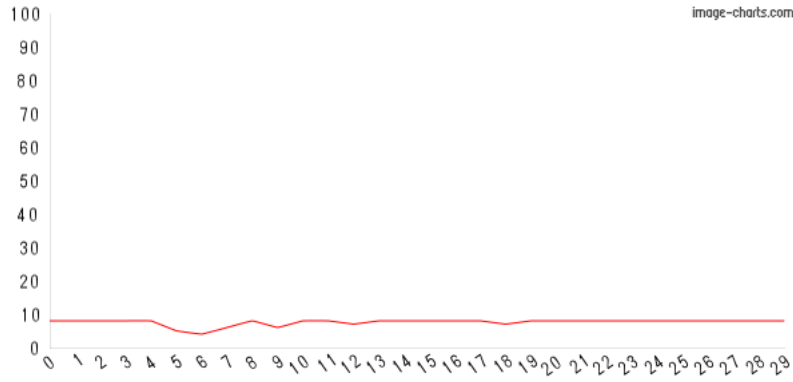


Figure 5: Gráfica de recompensas generada por el Benchmark

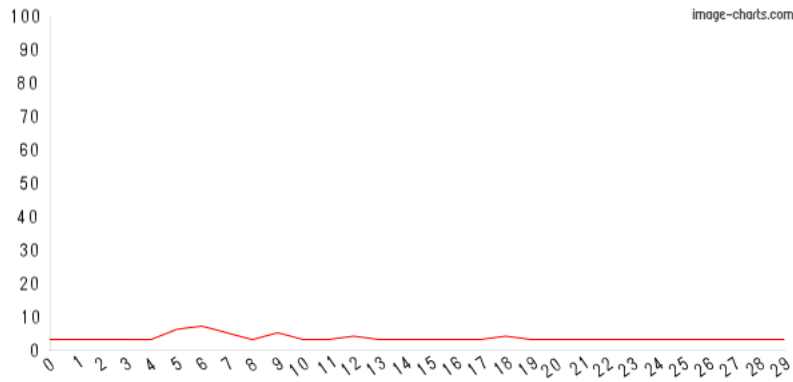


Figure 6: Gráfica de pasos generada por el Benchmark

En resumen, podríamos decir que teniendo en cuenta el reducido número de problemas con los que se ha probado cada algoritmo, cuando es un problema pequeño pero complejo, PBVI tiene mejores resultados. Pero cuando el problema es demasiado grande, es decir, que tiene muchas acciones posibles, como el caso del Laser Tag, conviene abordarlo mediante POMCP.

5 CONCLUSIONES

Este trabajo nos ha servido para ampliar mucho nuestros conocimientos de planificación, gracias en parte a la desviación inicial en la investigación, que nos hizo

profundizar en el ámbito de POMDP. Además de la notable inexperiencia trabajando con la tecnología propia de la librería escogida[7], lo que nos ha aportado un conocimiento muy útil tras solucionar errores en la misma y fallos de dependencias.

Tras el desarrollo del trabajo, después de la lectura de varios artículos, hemos descubierto que conforme se ha ido desarrollando la investigación a lo largo de los últimos años, este método se ha desmarcado de la línea principal comparándolo con otros más populares. Desarrollar el trabajo usando la librería, descifrando los entresijos del propio método, han servido para ver las limitaciones del mismo, pero también cuales son las situaciones más propicias para su uso.

Cabe destacar que no hay demasiadas fuentes disponibles en las que apoyarse, por lo que se ha colaborado con otros grupos[1] para el desarrollo de la investigación, y es una realizada que colaborando hemos sido capaces de llegar a muchas mas fuentes de información y a distintas tecnologías a usar, donde sendas valoraciones nos han aportado puntos de vista distintos, muy a tener en cuenta en éste, el campo de la investigación.

ANEXO I



Figure 7: Selecccion de problema

```
#####
#
# POMDP-IA
#
#####
# 1) POMCP
# 2) PBVI
#####
Elige un algoritmo :
```

Figure 8: Selección de Algoritmo

```
#####
#
# POMDP-IA
#
#####
# 1) Simulación Interactiva
# 2) Simulación Silenciosa
# 3) Benchmark
#####
Elige un modo de simulación :
```

Figure 9: Selección de modo de ejecución

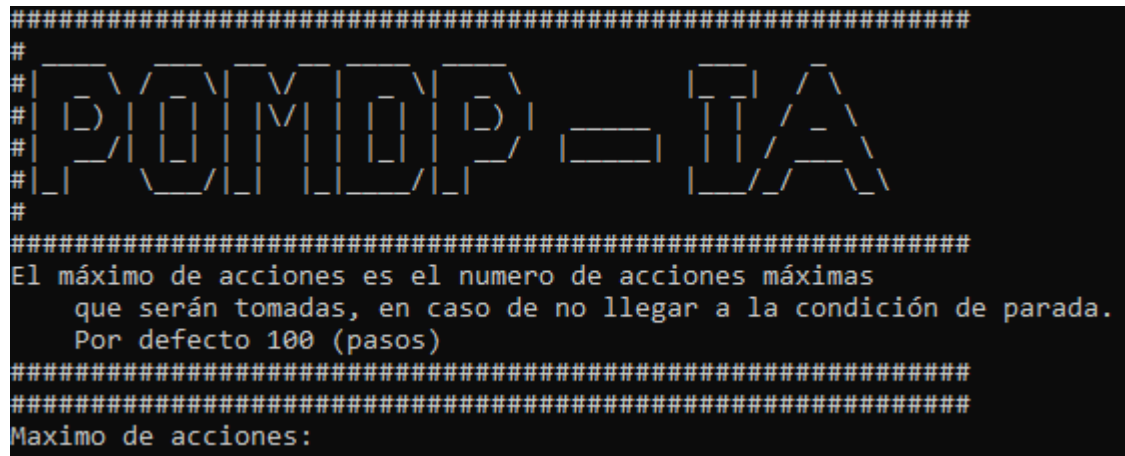


Figure 10: Selección de acciones máximas a realizar

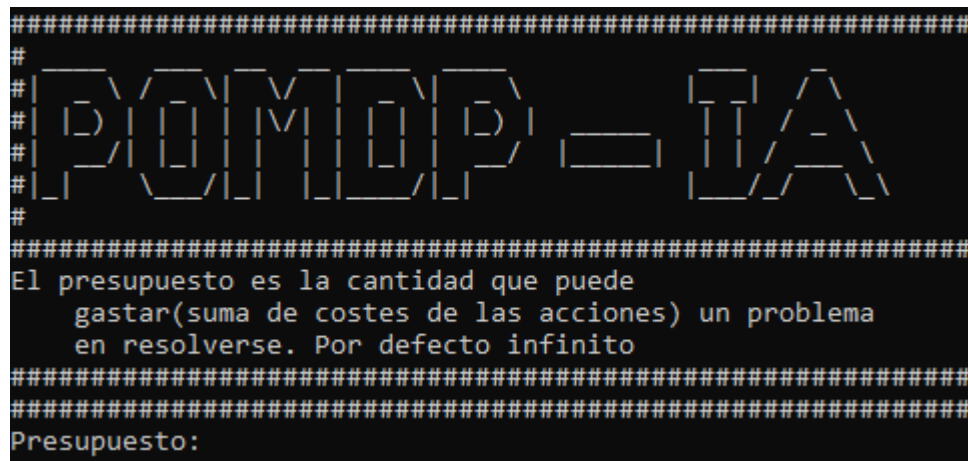


Figure 11: Selección de presupuesto

ANEXO II

La interfaz gráfica es un poco mas limitada que la terminal, algunas restricciones como bloquear PBVI para Laser TAG no están controladas. Aunque se ejecute mediante interfaz gráfica la salida sera mediante terminal. Al igual que las confirmaciones de ejecución.

```
$ python ./interfaz.py
```

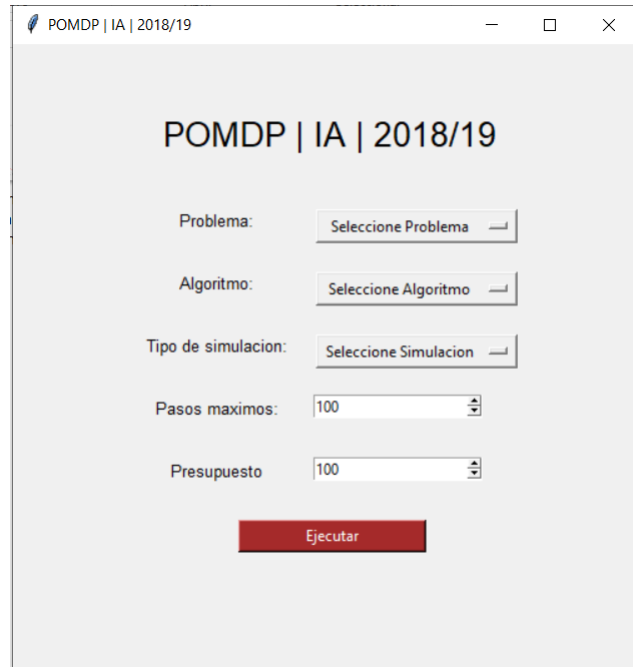


Figure 12: Interfaz Grafica

ANEXO III

El problema de Indiana Jones trata de este personaje de ficción que se encuentra en la siguiente situación de peligro. Esta encadenado en un pequeño patio donde existen 3 túneles. La cadena le permite refugiarse en uno de esos tres túneles, debe elegir uno para ello. Debido que una bola de piedra esta cayendo por uno de esos túneles, para decidir que túnel tomar puede poner la mano entre dos de ellos para notar la vibración pero esta percepción no es clara debido a los nervios y que la vibración se expande y llega a ambos lados aunque con distinta intensidad. Debe elegir un lugar antes de que la piedra llegue al patio(Pasos máximos)



Figure 13: Dibujo

Las acciones posibles son las siguientes:

- **apoyar-izquierda:** Apoyar la mano entre el túnel izquierdo y central.
- **apoyar-derecha:** Apoyar la mano entre el túnel derecho y central.
- **izquierda:** Elegir el túnel izquierdo para refugiarse.
- **centro:** Elegir el túnel central para refugiarse.
- **derecha:** Elegir el túnel derecho para refugiarse.

Los estados posibles son tres:

- **roca-izquierda:** La roca se aproxima por el túnel izquierdo.
- **roca-centro:** La roca se aproxima por el túnel central.
- **roca-derecha:** La roca se aproxima por el túnel derecho.

ANEXO IV

El problema de los anuncios trata sobre un sitio web que ofrece una gama de productos y desea adaptar los productos ofrecidos o anunciados al tipo de persona que está visitando su sitio. No quieres interferir con su navegación, por lo que las únicas pistas de su personalidad son productos en los que parecen estar interesados.[2]

Las acciones posibles son las siguientes:

- **A1:** Anunciar primer producto.
- **A2:** Anunciar segundo producto.
- **AN:** No anunciar.

Los estados posibles son tres:

- **S1:** Cliente tiene preferencia por el primer producto.
- **S2:** Cliente tiene preferencia por el segundo producto.
- **SN:** Cliente no tiene preferencia.
- **SX:** No hay clientes en el sistema.

REFERENCIAS

- [1] Colabroacion con el grupo formado con José Manuel Maestre Rodriguez y Lorezo Rondán Domínguez.
- [2] Pomdp examples. Disponible en: <http://www.pomdp.org/examples/>.
- [3] Pomdpdp. Disponible en: https://en.wikipedia.org/wiki/Partially_observable_Markov_decision_process.
- [4] Ignacio Perez Hurtado de Mendoza. Planificacion en entornos con incertidumbre. Disponible en: <https://www.cs.us.es/cursos/iais-2018/trabajos/pomdp.pdf>.
- [5] JL Iglesias Feria. Ia grafos - pomdp. Disponible en <https://www.youtube.com/watch?v=57njmzC3ksE>.
- [6] Geoff Gordon Joelle Pineau and Sebastian Thrun. Point-based value iteration: An anytime algorithm for pomdps. Disponible en: <http://www.cs.cmu.edu/~ggordon/jpineau-ggordon-thrun.ijcai03.pdf>.
- [7] namoshizun. Pypomdp. Disponible en <https://github.com/namoshizun/PyPOMDP>.
- [8] pemami4911. Pomdpy. Disponible en <https://github.com/pemami4911/POMDPy>.
- [9] David Silver Joel Veness. Monte-carlo planning in large pomdpss. Disponible en: <https://papers.nips.cc/paper/4031-monte-carlo-planning-in-large-pomdps.pdf>.