Implementación de Técnicas de Aprendizaje por Refuerzo

Fabián Levin [1], Sebastián Seba [1], Lucía B. Vallejos[1].

[1]Alumnos de la Cátedra de Inteligencia Artificial de 5º Nivel de I.S.I.
U.T.N., Facultad Regional de Resistencia – French 414 – Resistencia, Chaco, Argentina

Resumen. En el presente trabajo se describe un problema conformado por un espacio de estados en el cual se parte desde un estado inicial con el objetivo de alcanzar el estado final maximizando la recompensa que se obtiene. Para resolver este tipo de problemas se pueden utilizar diferentes técnicas de Inteligencia Artificial. El objetivo de este trabajo es describir un modelo para resolver el problema planteado a través de la utilización de Técnicas de Aprendizaje por Refuerzo (RL) mediante la implementación del algoritmo Q-Learning. A partir de esto, se analizan los resultados obtenidos con el objeto de establecer comparaciones de rendimiento entre las diferentes políticas de selección implementadas.

Palabras Claves: Aprendizaje por refuerzo, Q-Learning, ε-Greedy, Softmax.

1 Introducción

Existen gran cantidad de problemas complejos en los que resulta difícil para un ser humano lograr establecer una función de evaluación que le proporcione la mejor forma de resolverlos, a través de valoraciones precisas y consistentes para cada uno de los numerosos estados que los componen [1]. Los escenarios representados por espacios de estados son algunos de los problemas complejos que se mencionan anteriormente, debido a que a cada paso de su resolución es necesario considerar una gran cantidad de factores que al final permitan encontrar una solución óptima. Considerando la "explosión combinatoria" que se puede presentar, las pruebas exhaustivas resultan totalmente ineficientes. Por lo tanto, en dominios complejos, Aprendizaje por Refuerzo (RL, del inglés Reinforcement Learning) es la única forma factible de entrenar un programa para que su desempeño sea de alto nivel [1].

Las técnicas de RL son adecuadas para resolver problemas asociados a entornos dinámicos e impredecibles y se aplican en algunos campos como la robótica y videojuegos.

En este trabajo se analiza un modelo de resolución para el problema planteado aplicando técnicas de RL con el objetivo de estimar una política óptima para resolverlo, es decir aquella que proporcione una solución que maximice los beneficios obteni-

dos. Existen trabajos similares desarrollados aplicando otras técnicas como las técnicas de Búsqueda Heurística [2].

En la sección dos se realiza una introducción a los conceptos básicos de RL y en particular de Q-Learning. Luego, en la sección tres, se describe el problema planteado seguido de la solución que se implementa, la cual se encuentra en la sección cuatro. Por otra parte, en la sección cinco se presentan los problemas hallados, en la sección seis se detallan las simulaciones realizadas y la última corresponde a las conclusiones del trabajo.

2 Aprendizaje por Refuerzo y Q-Learning

RL es un paradigma de aprendizaje relacionado con el proceso de aprendizaje para controlar un sistema con el fin de maximizar una medida de rendimiento que expresa un objetivo planteado [3].

A diferencia del aprendizaje supervisado donde existe un agente externo que provee de un conjunto de entrenamiento al sistema, en las técnicas de RL el agente debe interactuar con un entorno complejo e incierto, perfeccionando su comportamiento según los estímulos que recibe del ambiente a cada acción realizada. Por lo tanto, en estas técnicas no existe una supervisión externa, sino un proceso de retroalimentación continuo entre el agente y el entorno durante todo el aprendizaje.

Se puede relacionar a RL con la ciencia de la conducta adaptativa de los seres racionales en entornos desconocidos [3].

Los problemas tratados mediante RL pueden ser determinísticos o no determinísticos. Aquellos problemas en los que existe un final u objetivo al cual se desea llegar, es decir son finitos, y además las variables que representan al dominio del problema son constantes, son los llamados problemas determinísticos. Los problemas no determinísticos resultan más complejos para resolverlos ya que se caracterizan por poseer variables con comportamientos probabilísticos, las cuales resultan imposibles de determinar con exactitud.

Q-Learning es una implementación de RL desarrollada tanto para problemas determinísticos como no determinísticos [4] aplicada para resolver el problema que se plantea en este trabajo, el cual es de tipo determinístico.

El algoritmo de Q-Learning se define mediante la siguiente ecuación:

$$(\quad) \quad (\quad) \qquad (\quad) (\quad) \qquad (1)$$

Donde *s* es el estado actual, *a* es la acción elegida a partir del estado *s*, *s'* es el estado siguiente a *s* ejecutando la acción *a*, *a`* corresponde a todas las acciones posibles a partir del estado *s'*, γ es el factor de aprendizaje el cual determina el valor presente de una recompensa futura variando dicho factor entre 0 y 1, y *R* es la recompensa inmediata, obtenida a partir de la *matriz R*, correspondiente al par estado-acción.

Esta fórmula representa la actualización de un valor asociado a un elemento perteneciente a una matriz, denominada *matriz Q,* la cual constituye las recompensas a largo plazo de cada par estado - acción.

Las políticas o estrategias de selección de la acción que se utilizan en este trabajo son ε-Greedy y Softmax, las cuales se detallan a continuación.

La política ε-Greedy, utiliza un parámetro ε el cual representa la probabilidad de explorar. El balance entre la exploración y explotación se obtiene mediante la variación de este parámetro entre el intervalo (0,1). En la exploración se elige la acción de forma uniformemente aleatoria y en la explotación se selecciona la acción que posee mayor valor Q. En ambos casos se selecciona una de entre todas las acciones posibles a partir del estado actual.

Softmax, por otra parte, selecciona una acción con probabilidad:

$$( ) \quad \frac{\overline{\quad ( ) \quad}}{\Sigma \ \overline{\quad ( ) \quad}} \tag{2}$$

Donde τ es un parámetro positivo denominado *temperatura*. Un valor alto de τ produce que las acciones tiendan a ser equiprobables mientras que a valores pequeños de τ, las acciones con mayores valores de Q, poseen mayores probabilidades.

En el presente trabajo se evalúa el proceso de aprendizaje utilizando una de las fórmulas aplicadas en RL para la evaluación del aprendizaje [4] la cual corresponde al cálculo de recompensas promedios para cada acción por episodios:

$$( ) \quad \overline{\qquad\qquad} \tag{3}$$

Donde representan las recompensas inmediatas desde el estado 1 hasta el estado *k*-ésimo asociadas a la acción *a,* correspondientes al episodio del tiempo *t*.

3 Descripción del Problema

El problema que se presenta consta de un espacio de estados conformado por una grilla de dimensión cuadrada que puede variar de tamaño de orden 6 a 10. Ésta puede poseer diferentes configuraciones (determinadas por tipos de estados y dimensiones dadas) a elección del usuario o generadas aleatoriamente.

Los tipos de estados con los que se puede configurar la grilla son: inicial, final, excelente, bueno, malo, neutro y pared. Cada configuración debe poseer un único estado inicial y final mientras que con respecto al resto de los tipos de estados la cantidad puede ser arbitraria. Una consideración específica en cuanto al tipo de estado pared es que el agente debe evitar pasar por ellos ya que al hacerlo "muere", lo que significa que debe recomenzar el juego desde el estado inicial.

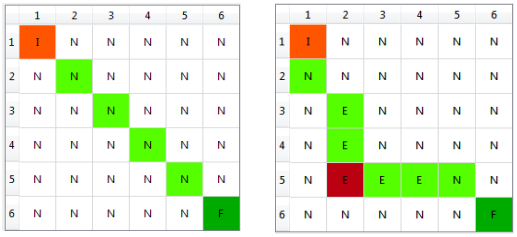
Los movimientos posibles del agente son como máximo ocho, siempre y cuando no se ubique en una esquina o borde de la grilla. Por lo tanto, sólo puede moverse a casilleros contiguos al actual.

El objetivo consiste en que el agente, partiendo desde el estado inicial, pueda llegar al estado final por el camino óptimo, el cual dependerá de los criterios tomados para la resolución del problema.

4 Descripción de la Solución Implementada

En primer lugar es necesario establecer los criterios que se tienen en cuenta para definir la solución óptima en cada configuración de la grilla.

Se define el camino óptimo como aquel determinado por un balance entre el camino más corto y los casilleros que otorguen mayores recompensas. Es decir que, en circunstancias donde las recompensas otorgadas por estados cercanos al camino más corto son considerables, el agente se desvía tomando un camino con mayor cantidad de pasos hasta el final pero con recompensas mayores. (ver Figura 1).



(a) **(b)**

Fig. 1. Soluciones óptimas **(a)** Sin casilleros excelentes, **(b)** Con casilleros excelentes

Teniendo en cuenta los criterios anteriormente mencionados para definir el camino óptimo se establecen los valores de las recompensas de los distintos tipos de estados presentes en el escenario del problema (ver Tabla 1).

Tabla 1. Valores de Recompensas para cada tipo de estado.

Estados	Malo	Neutro	Bueno	Excelente	Final	Pared
Recompensas	-50	0	50	100	Variable	-

En cuanto a los estados malo, neutro, bueno y excelente las recompensas fueron establecidas a partir de criterios relativos definidos entre ellos mismos como que el estado malo debe ser peor que el neutro, el bueno mejor que el neutro, y el excelente mejor que el bueno.

Para el caso del estado final, el valor de la recompensa es variable lo cual no significa que su valor varíe durante el proceso de aprendizaje sino que se calcula mediante parámetros establecidos antes de cada entrenamiento. Esto es necesario para lograr que el comportamiento del agente sea acorde con el camino óptimo planteado.

Para definir una expresión que permita establecer los valores adecuados de la recompensa del estado final, se parte del análisis de la ecuación (1) presentada en la sección 2. Teniendo en cuenta el segundo miembro de dicha ecuación, se determina que una recompensa recibida k pasos después, sólo vale veces de lo que valdría si fuese inmediata [4]. Esto significa que, a una distancia de n pasos del estado final, la recompensa otorgada por éste queda determinada por el producto entre el factor de aprendizaje elevado a $n-1$ y el valor de la recompensa del estado final:

(4)

Donde representa la recompensa que otorga el valor de la recompensa del estado final *F,* dada una distancia de *n* pasos, en un estado *i*. A partir de (4) se puede deducir que, a mayor cantidad de pasos *n* entre el estado inicial y final y valores menores del factor de aprendizaje, resulta necesario que el valor de la recompensa del estado final sea mayor. Si esto se cumple, a pesar de que el agente se encuentre en casilleros lejanos al estado final, la recompensa que éste otorga a futuro resulta considerable para el agente, logrando así que pueda llegar a dicho estado final.

Es necesario establecer el valor de recompensa del estado final en forma relativa a la recompensa del estado excelente, debido a que éste es el siguiente tipo de estado con mayor valor de recompensa. Por lo tanto, si se determina el valor de recompensa del estado final que aporte al agente beneficios más significativos que los estados excelentes, cualquiera sea la configuración de la grilla, el valor de la recompensa del estado final será también mejor que los valores de los demás tipos de estados que poseen recompensas menores. De esta manera, la recompensa otorgada por el estado final (*F*) dada una distancia de *n* pasos considerada, debe ser igual (o mayor) a la recompensa otorgada por el estado excelente (*E*):

(5)

Finalmente, despejando de la ecuación (5) se obtiene la expresión para el cálculo de la recompensa del estado final, dada por:

—— (6)

Se establecen de forma aproximada los valores *n* de los cuales resultan las recompensas del final más adecuadas para cada tamaño de la grilla (ver Tabla 2).

Tabla 2. Valores de *n* para cada dimensión de la grilla

Dimensión de la grilla	6x6	7x7	8x8	9x9	10x10
Valor de n	13	18	20	29	32

Por último, el tipo de estado pared no posee recompensa ya que se ha planteado como un estado al cual el agente no puede acceder. De esta manera se solucionan los problemas hallados que se describen en la Sección 5.

Con respecto a la matriz Q se implementan dos formas de inicializarla antes del proceso de aprendizaje: con todos sus valores en cero o utilizando la técnica de Valores Optimistas. Ésta consiste en inicializarla con valores que sean mayores a la recompensa más alta.

Concretamente, para implementar la solución propuesta se ha decidido utilizar Python [5] como lenguaje de programación principal complementado con el paquete para computación científica NumPy [6] y la biblioteca para generación de gráficos Matplotlib [7].

5 Descripción de los Problemas Hallados

La mayor parte de los problemas hallados surgen en cuanto a las definiciones de las distintas recompensas de los estados. En primera instancia, se plantea como camino óptimo aquel en el cual el agente llegue al final luego de recorrer la mayor cantidad posible de estados excelentes o buenos. Para lograr esto es necesario que la recompensa del estado final sea considerablemente menor a la que se establece finalmente. Esto ocasiona una serie de problemas que se describen a continuación.

En configuraciones donde existen varios estados excelentes o buenos contiguos entre sí, este conjunto constituye una zona que posee mayor valor con respecto al estado final. Esto ocasiona un problema al cual se le da el nombre de "Zona de Confort" haciendo analogía a que el agente es "atraído" hacia dicha zona y, una vez allí, se mantiene recorriendo este grupo de estados excelentes o buenos, lo que imposibilita que llegue al estado final. Se intenta resolver esta cuestión aplicando un control de bucles en el cual se "marcan" los estados ya recorridos, lo que evita que el agente pueda volver a pasar por ellos. Sin embargo, esto ocasiona otro problema: en aquellas configuraciones con paredes sucede que el agente queda encerrado sin llegar al estado final a pesar de que existe un camino hacia el mismo. Además, debido a que el comportamiento del agente está condicionado casi en igual medida por los estados con recompensas buenas como por el estado final, ocurren constantemente movimientos sin sentidos como recorrer varias veces un mismo casillero, moverse en un sentido y dirección y luego cambiar hacia otros contrarios y que no están dirigidos hacia el estado final, etc.

Como consecuencia de los problemas descriptos anteriormente, una vez que se analizan, se logran definir los criterios que permiten solucionarlos. Sin embargo, se detectan configuraciones en las que, habiendo un camino posible, el agente no llega al estado final. En aquellos casos donde se consideran distancias de mayor cantidad de pasos entre el estado inicial y el final, que las distancias en las configuraciones consideradas anteriormente para definir la recompensa del estado final, esta recompensa no es lo suficientemente grande para ser percibida por el agente a dicha mayor cantidad de pasos y lograr que pueda llegar al estado final. También se produce este problema mencionado cuando, además de considerar grandes distancias entre los estados inicial y final, existen estados excelentes cerca del inicial y/o malos cerca del final, que opacan al valor de éste último, por ejemplo la configuración mostrada en la Figura 2.

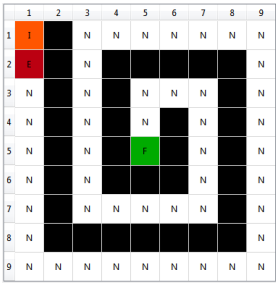


Fig. 2. Ejemplo de Configuración con problemas detectados

Se determina que los valores adecuados para las recompensas del estado final que permitan hallar soluciones en las configuraciones mencionadas previamente, son significativamente mayores en relación a los que se establecen anteriormente, por lo tanto, el comportamiento del agente cambia por completo. Esto implica que dicho agente ya no determina una solución que haga un balance entre el camino con mayores casilleros excelentes y buenos y el camino más corto. En todos los casos sigue la opción del camino más corto. Es decir, los casilleros excelentes y buenos no son tomados en cuenta por el agente debido al gran valor de la recompensa del estado final, que provoca que sólo busque llegar a éste. Para la solución planteada se considera que si las recompensas de todos los tipos de estados, a excepción del estado final, son despreciables por el agente y las soluciones que éste determina, no tiene sentido definir dichos estados. Por lo tanto, se opta por definir a estas configuraciones donde se hallan problemas, como casos excepcionales sin solución con el objeto de mantener los criterios que se definen para el comportamiento del agente.

Por otra parte, con respecto también a la solución implementada, se identifican problemas asociados a la fórmula utilizada en el cálculo de los valores de la recompensa del estado final (6). Dichos problemas ocurren para valores pequeños del factor de aprendizaje que resultan en valores muy grandes para la recompensa del estado final y que luego provocan problemas de desbordamiento de datos en los cálculos exponenciales correspondientes a la política Softmax. Debido a que este tipo de problema está ligado a la tecnología utilizada, la solución que se encuentra es limitar el ingreso por interfaz gráfica de los valores para el parámetro del factor de aprendizaje calculando en forma automática los valores mínimos permitidos.

Otro problema surge en la definición del tipo de estado pared. Como se explica anteriormente éste se define como estado inaccesible, lo cual significa que es descartado de los casilleros posibles a donde el agente puede moverse. Pueden existir casos donde no hay solución, es decir, todos los caminos posibles desde el estado inicial hacia el estado final se encuentran bloqueados por paredes. Sin embargo, si se permite al agente acceder a los tipos de estados pared, aunque obtenga un castigo por ello (por ejemplo, una recompensa de valor negativo), lo hace de todas maneras tomando el camino con el menor castigo. Esto provoca una acción que es arbitraria para el comportamiento del agente, es decir, que pueda pasar a través de estados pared. Por lo tanto, al definir los estados pared como estados inaccesibles, se evita este inconveniente y aquellas configuraciones de la grilla que no tengan solución, son identificadas por un control de bucle como casos sin solución y se abortan.

6 Simulaciones y Resultados

El objetivo, con respecto a las simulaciones, es poder evaluar la eficiencia en el proceso de aprendizaje según los parámetros de entrada, las políticas de selección de acciones utilizadas y las configuraciones de la grilla. Para ello se organizan las pruebas centrándose en cada uno de estos aspectos mencionados. En primer lugar, durante el proceso de evaluación, se realizan las pruebas definiendo los conjuntos de parámetros con valores bajos, medios y altos de manera que se abarque, en general, el domi-

nio de valores de cada variable. Luego se ejecutan nuevas pruebas centrándose en aquellas configuraciones que arrojan mejores resultados.

A partir de la ecuación para la evaluación del aprendizaje (3) presentada en la sección 2, la cual expresa la recompensa promedio para cada acción por episodios, se toma el promedio de todas ellas por cada episodio como medida para analizar los resultados de las simulaciones.

6.1 Evaluación respecto al Nivel de Dificultad de las Configuraciones

Con el objeto de evaluar si el aprendizaje del agente se ve afectado por el nivel de dificultad que presentan las configuraciones, se establecen 3 grillas con distintos niveles de dificultad: baja, media y alta. Para ello se tienen en cuenta las complicaciones presentes en dichas configuraciones como: mayor cantidad de paredes, mayor distancia desde el estado inicial al final, entre otras cosas.

Los resultados que se obtienen, teniendo iguales parámetros de entrada y diferentes niveles de dificultades, se comparan observando que presentan características similares y permiten llegar las mismas conclusiones. Por lo tanto, en cuanto a estos gráficos, sólo se muestran los resultados para el nivel de dificultad baja. Esto permite concluir que la eficiencia del aprendizaje no se ve afectada por la dificultad de las configuraciones. Sin embargo, al comparar el porcentaje de episodios en los que el agente llega al final sobre el total de episodios, cuando se limita la cantidad de iteraciones o movimientos por episodio, se observa que a mayor nivel de dificultad de la grilla este porcentaje disminuye (ver Figura 3).

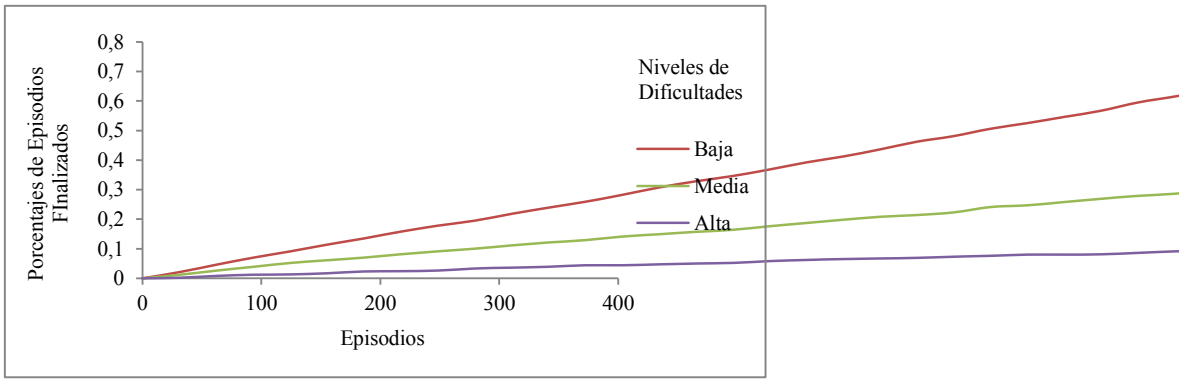


Fig. 3. Porcentajes de episodios finalizados para distintos niveles de dificultad.

6.2 Evaluación respecto al Factor de Aprendizaje

En las pruebas, donde se varían sólo los valores del factor de aprendizaje, manteniendo constantes los demás parámetros, se obtienen los mismos resultados en cuanto a la eficiencia del aprendizaje. Esto ocurre debido a la ecuación que se utiliza en este trabajo para calcular el valor de la recompensa del estado final (6) que se desarrolla en la sección 4, la cual adapta el valor de dicho estado según el valor del factor de aprendi-

zaje. Con estos resultados se puede concluir que se logra que el agente, para configuraciones iguales, pueda aprender con la misma eficiencia inclusive asignando menores valores al factor de aprendizaje.

6.3 Evaluación de la Política de Selección de acciones ε-Greedy

Para evaluar la eficiencia del aprendizaje, al aplicar la técnica de selección de acciones ε-Greedy, explicada en la sección 2, se realizan pruebas variando los parámetros de ε. Los resultados obtenidos muestran que la curva de aprendizaje tiende a alcanzar mayores valores en episodios más tempranos, y por lo tanto a estabilizarse más rápido, cuanto más pequeño es el valor de ε (como se muestra en la Figura 4). Esto significa que la eficiencia en el aprendizaje es mayor a medida que disminuye el valor en la probabilidad de la acción explorar.

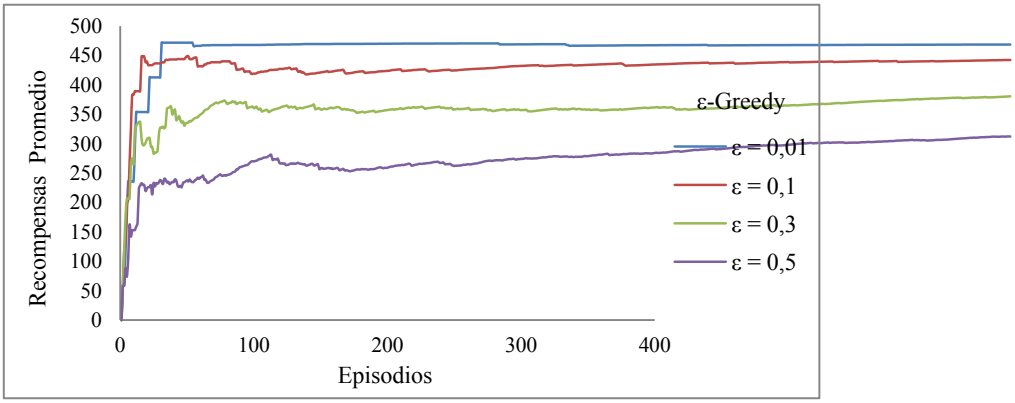


Fig. 4. Evaluación de ε-Greedy para distintos valores de ε.

Por otra parte, se realizan las mismas pruebas utilizando la técnica de Valores Optimistas, detallada en la sección 2, con el objeto de observar su efecto en la eficiencia del proceso de aprendizaje. Como resultado se observa que el aprendizaje mejora, mostrando valores más altos en los episodios iniciales en comparación a los resultados obtenidos en las pruebas anteriores, en las cuales se inicializa la matriz Q en cero (ver Figura 5).

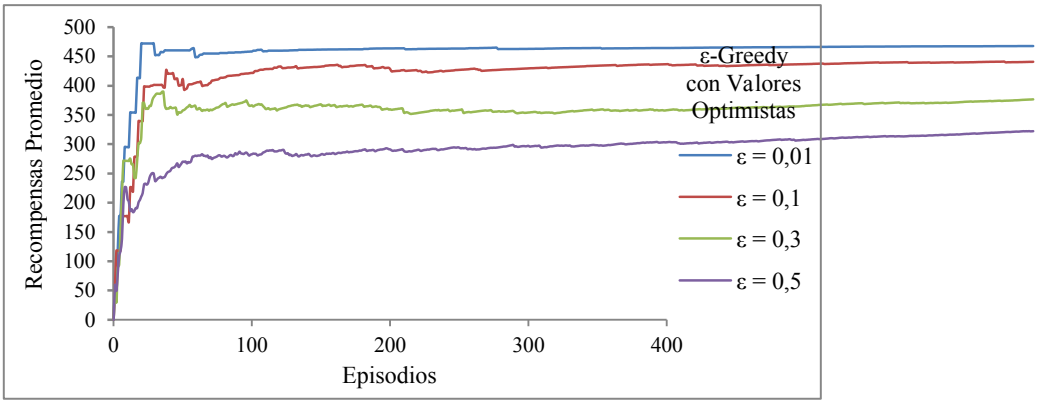


Fig. 5. Evaluación de ε-Greedy con Valores Optimistas para distintos valores ε.

6.4 Evaluación de la Política de Selección de acciones Softmax

Para la evaluación de la eficiencia del aprendizaje en relación a la política Softmax, detallada en la sección 2, se realizan las mismas pruebas para distintos valores de τ. Los resultados son similares a los obtenidos con la política ε-Greedy, ya que se observan mejores resultados para valores de τ más pequeños (ver Figura 6), como también un aumento en la curva de aprendizaje en los episodios iniciales al aplicar la técnica de Valores Optimistas (ver Figura 7).

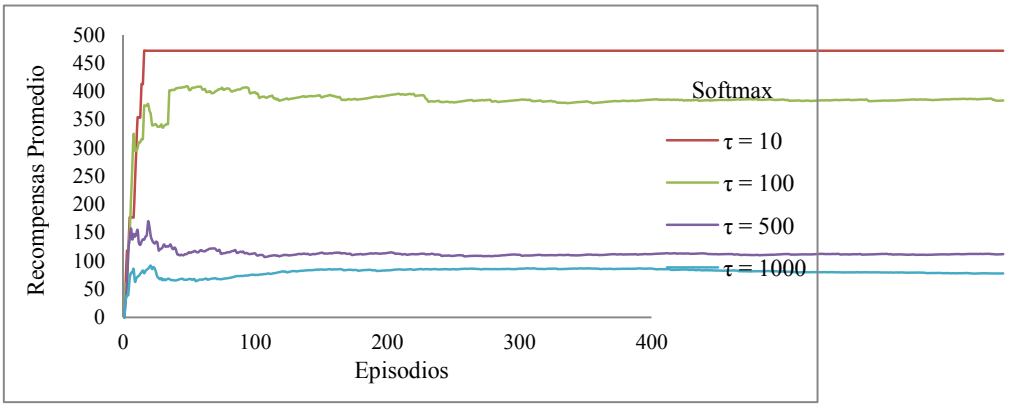


Fig. 6. Evaluación de Softmax para distintos valores de τ.

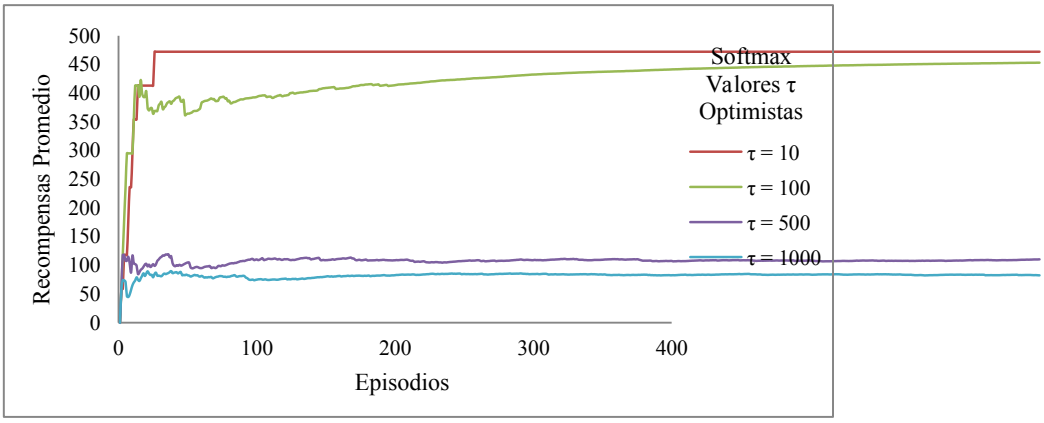


Fig. 7. Evaluación de Softmax con Valores Optimistas para distintos valores de τ.

6.5 Comparación entre las Políticas de Selección de acciones ε-Greedy y Softmax

Por último, se realiza una comparación entre los mejores resultados obtenidos en las evaluaciones de las políticas ε-Greedy y Softmax para lograr establecer cuál de ellas alcanza mayores niveles de eficiencia. Se observa que ambas logran alcanzar niveles de eficiencia similares en el aprendizaje con una pequeña mejora por parte de Softmax (ver Figura 8).

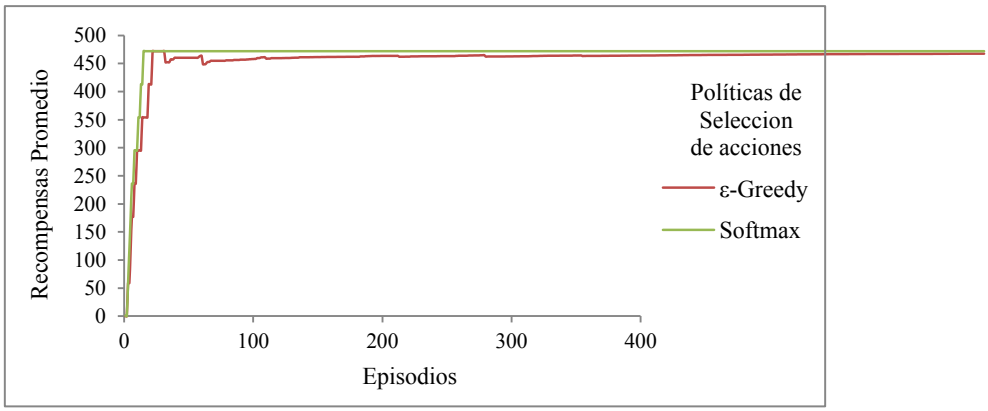


Fig. 8. Comparación de ε-Greedy y Softmax

Como resultado de estas corridas explicadas entre las subsecciones 6.1 a 6.5, se eligen las configuraciones que proporcionan los resultados más satisfactorios para establecerlas como valores por defecto en la aplicación desarrollada.

7 Conclusiones

Los resultados que se obtienen en las simulaciones demuestran que la aplicación de técnicas de RL, para la solución de este tipo de problemas, puede permitir alcanzar altos niveles de eficiencia siempre que se utilicen los valores adecuados en los distintos parámetros. Con respecto a la política de selección de acciones ε-Greedy, a valores más pequeños para el parámetro ε (próximos a 0.01), se obtienen mayores niveles de eficiencia en el proceso de aprendizaje. De igual manera sucede con Softmax, donde a menores valores de τ (próximos a 10) se observa un aumento en la eficiencia del aprendizaje. Se puede apreciar que ambas políticas alcanzan altos niveles de eficiencia para los parámetros mencionados, superando Softmax a ε-Greedy en una pequeña medida. Además en ambas se observa una mejora en la eficiencia al aplicar la técnica de Valores Optimistas en la inicialización de la matriz Q.

El aspecto más importante en el desarrollo de modelos de solución utilizando éstas técnicas es la definición de las recompensas. Se puede observar que el comportamiento del agente está fuertemente condicionado por los valores de dichas recompensas debido a la característica del aprendizaje basado en las interacciones con el entorno, por lo tanto es importante adecuar esos estímulos para lograr el comportamiento deseado.

Una desventaja referida a la aplicación de las técnicas de RL en este problema específico, es que no es posible encontrar solución en todos los casos, ya que en aquellas configuraciones con alto grado de complicaciones resulta más difícil que el agente pueda hallar una solución. A pesar de que ésto sólo sucede en casos particulares, se pueden hallar soluciones a través de la aplicación de otras técnicas como las correspondientes a los algoritmos de búsqueda sistemática o heurística. Sin embargo, frente a pruebas exhaustivas que provocan una "explosión combinatoria" en la resolución de este tipo de problemas, la técnica de RL arroja resultados óptimos aprovechando los recursos de manera más eficiente.

Referencias

1. S. Russell and P. Norvig: Artificial Intelligence, A modern Approach. 3 rd Edition. Pearson Prentice Hall (2010).
2. O. Fernandez, L. Felice: Algoritmos de búsqueda heurística en tiempo real, Aplicación a la navegación en los juegos de video. XXXIV Jornadas Argentinas de Informática e Investigación Operativa. Rosario,Santa Fé (2005).
3. C. Szepesvári: Algorithms for Reinforcement Learning, Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers (2010).
4. R. Sutton, A. Barto: Reinforcement Learning, An Introduction. MIT Press (1998).
5. Python Programming Language; http://www.python.org
6. Numerical Python; http://numpy.org
7. Matplotlib: A 2D graphics environment; http://matplotlib.org