UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MATEMÁTICA

DESARROLLO DE UN ALGORITMO DE ASIGNACIÓN AUTOMÁTICA DE EJERCICIOS COMO HERRAMIENTA DE APRENDIZAJE, BASADA EN EL DESEMPEÑO DE LOS ESTUDIANTES.

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERIA, MENCIÓN MATEMÁTICAS APLICADAS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO

PEDRO IGNACIO VERGARA IGLESIAS

PROFESOR GUÍA:
PABLO DARTNELL ROY

PROFESOR CO-GUÍA:
CRISTIAN REYES REYES

COMISIÓN:
PATRICIO FELMER AICHELE

Este trabajo ha sido parcialmente financiado por CMM ANID PIA AFB170001

SANTIAGO DE CHILE
ENERO 2021

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO
TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN
CIENCIAS DE LA INGENIERIA, MENCIÓN MATEMATICAS APLICADAS
POR: PEDRO IGNACIO VERGARA IGLESIAS
FECHA: ENERO 2021
PROF. GUÍA: PABLO DARTNELL ROY

DESARROLLO DE UN ALGORITMO DE ASIGNACIÓN AUTOMÁTICA DE EJERCICIOS COMO HERRAMIENTA DE APRENDIZAJE, BASADA EN EL DESEMPEÑO DE LOS ESTUDIANTES.

La educación es uno de los pilares fundamentales en la construcción de una sociedad moderna, y es necesaria una constante adaptación de las metodologías de enseñanza, en particular, la introducción de nuevas herramientas tecnológicas que se enfoquen en complementar y profundizar el proceso de aprendizaje dentro y fuera del aula de clases, es de vital importancia para llevar a cabo el proceso de enseñanza en la era digital. En los últimos años, los avances en técnicas de *Machine Learning* han permito el desarrollo de soluciones de aprendizaje adaptativo para reforzar los contenidos vistos en clases. Estas soluciones han tomado forma como aplicaciones web y softwares educativos que buscan apoyar el estudio personal de un alumno, en base a su interacción con una variedad de elementos tales como ejercicios, videos, pruebas estandarizadas, juegos, etc. Un proceso clave en el desarrollo de estas plataformas es la inclusión de algoritmos que tomen en cuenta el rendimiento personal de un alumno y que permitan introducir estrategias personalizadas que resulten idóneas para reforzar y/o potenciar las debilidades y fortalezas del estudiante.

Esta tesis esta compuestas por 5 capítulos, y se enfoco en la realización de un algoritmo que se adaptara al desempeño de un estudiante para desplegar en cada momento un ejercicio de dificultad acorde, utilizando la técnica de Machine Learning correspondiente al *Reinforcement Learning*. De igual manera, se estudiaron los principales conceptos teóricos relativos a los método de Reinforcement Learning que son *Programación Dinámica* y *Monte Carlo*. Además, puesto que no se contaba con datos de estudiantes reales para monitorear la efectividad del algoritmo trabajado, es que se ideó un simulador de estudiante.

En el capitulo 1 se introduce en qué consisten a modo general los métodos de Programación Dinámica y Monte Carlo. En el capítulo 2 se presenta el trabajo, objetivos de este, definiciones preliminares necesarias para entender los métodos utilizados, y el modelamiento del problema aterrizándolo a estos métodos. Luego en el capítulo 3 se comienza a trabajar y desarrollar téoricamente las fórmulas presentadas en el capítulo anterior para los métodos mencionados anteriormente de Reinforcement Learning, y se desarrolla una extensión de la teoría estándar. Esto permite poder dejar expresiones programables y corroborar teóricamente que estas fórmulas permiten encontrar lo que se busca. En el capítulo 4, se presentan resultados respecto al comportamiento del simulador, para luego presentar los resultados principales, correspondientes al algoritmo de decisión, variando los parámetros para presentar distintos casos de interés usando el método de Programación Dinámica o el de Monte Carlo. Finalmente, en el capítulo 5, corresponde a la sección de análisis y discusión de los resultados presentados anteriormente, comparando ambos métodos, y la complejidad asociada a estos respecto a los parámetros asociados.

Tabla de Contenido

Introducción

La educación es uno de los pilares fundamentales en la construcción de la sociedad moderna, la cual, al estar en constante cambio, repercute en los nuevos requerimientos en miras al futuro, adaptando las metodologías de enseñanza a los nuevos escenarios, donde serán protagonistas las nuevas generaciones. Con esto, y el uso de nuevas tecnologías se hacen más factibles metodologías que, anteriormente, sonaban utópicas.

En Chile existe una alta tasa de cantidad de estudiantes por docente. Por ejemplo, en el año 2016, el promedio de estudiantes por docente de enseñanza básica -en establecimientos particulares subvencionados de la RM- fue de 30 (ver [19] p. 68 y 134), evidenciando la necesidad de contar con nuevas herramientas tecnológicas, enfocadas en complementar y profundizar el proceso de aprendizaje tanto dentro, como fuera del aula de clases, entrando en el proceso de enseñanza en la era digital.

El uso efectivo de herramientas tecnológicas conlleva múltiples beneficios tales como acceso a contenido adaptativo, feedback inmediato, visualización de estadísticas, entre otros. Más aún, el empleo de computadores u aplicaciones web entrega a los estudiantes práctica en contenidos y habilidades base, permitiendo a profesores y docentes enfocar su labor de manera individualizada, facilitando una mejor diferenciación, lo que a su vez estimula que los estudiantes encuentren un ritmo de estudio personalizado (ver [10] y [15]).

En los últimos años, los avances en técnicas de Machine Learning han permitido el desarrollo de soluciones de aprendizaje adaptativo para reforzar los contenidos vistos en clases, esto lo entendemos como el desarrollo de algoritmos que tomen en cuenta el rendimiento personal de un alumno y que permitan introducir estrategias personalizadas que resulten idóneas para reforzar o potenciar las debilidades y fortalezas del estudiante. Estas soluciones han tomado forma como aplicaciones web y softwares educativos que buscan apoyar el estudio personal de un alumno. Ejemplos de estos desarrollos, en trabajos recientes se pueden encontrar:

- Plataformas para evaluaciones en línea basadas en métodos de predicción de desempeño por ítem o ejercicio, a partir de parámetros de habilidad personal (ver [18]).
- Plataformas de educación enfocadas en el aprendizaje de idiomas, tales como Duolingo, basada en el uso de regresiones para el cálculo de flujo de ejercicios en actividades de reforzamiento (ver [17]), a partir de la incorporación de un factor de memoria temporal.

El objetivo de este trabajo corresponde al desarrollo de una herramienta tecnológica de aprendizaje adaptativo, que utilice el desempeño de un estudiante para presentarle a resolver en cada momento ejercicios adecuados para su aprendizaje, es decir, que le sean desafiantes, pero de un nivel de dificultad que aborde conocimientos con los que ya cuentan. Esto se buscará lograr basado en la interacción entre un estudiante y un flujo de ejercicios que permita incluir de manera activa el rendimiento personal del estudiante.

Para intentar lograr este objetivo, se exploraron técnicas de Machine Learning (ver [21] y [11]), escogiendo entre estas y ahondando en los modelos de Reinforcement Learning (ver [20] p. 73-114), debido a que por la naturaleza teórica permite una adaptación natural al problema. Paralelamente se estudió la formalización estándar de los conceptos empleados, y formalizó la extensión propuesta, haciendo uso de técnicas asociadas a procesos estocásticos tales como *Markov Decision Process* (ver [2, 8], y [12] p. 1-9, 43-153).

Se establecieron criterios para maximizar el aprendizaje a través de una disposición adecuada de los elementos con los que interactúa el estudiante, en específico se desarrolló un algoritmo que optimiza el flujo de ejercicios en función del rendimiento de un estudiante, lo cual se especificará más adelante en esta tesis. Además se realizaron simulaciones del algoritmo desarrollado, con la perspectiva de medir el rendimiento e impacto en el apoyo a actividades de docencia.

Capítulo 1

Marco Teórico

Para el desarrollo del algoritmo que nos permita presentar un flujo adecuado de ejercicios a los estudiantes, con un nivel adaptado al desempeño individual, se exploró en las técnicas de Machine Learning, más precisamente, en los modelos asociados al Reinforcement Learning, deteniéndose principalmente en dos, que son el que utiliza técnicas de Programación Dinámica, y el que utiliza técnicas de Monte Carlo. A continuación hablaremos más sobre estos dos métodos con los que se trabajaron.

1.1. Método de Programación Dinámica

La Programación Dinámica es un método de optimización matemática desarrollado por Richard Bellman en 1953 (ver [6]). A grandes rasgos se refiere a la resolución de un problema complejo por medio de su descomposición en problemas más simples de manera recursiva.

En el contexto de optimización, se busca simplificar una decisión descomponiéndola en una secuencia de decisiones a lo largo del tiempo (o en sub problemas de optimización). Para esto se define una secuencia de **funciones de valor** $V_1, \ldots, V_n$ que toman como parámetros, elementos de un conjunto S que llamaremos **estados**, y los índices de estas funciones, corresponderán al tiempo discreto en que aparecen o se calculan (la función V_{i}, es la función de valor en el tiempo i). De modo que, $V_n(s)$ corresponde al valor obtenido en el estado **s** a tiempo **n**, y los valores anteriores V_{i} con $\mathrm{i} = n-1, \ldots, 1$ se calculan por medio de una ecuación recursiva denominada **Ecuación de Bellman**.

1.1.1. Ecuación de Bellman

Sea s_t el estado del sistema a tiempo t (elemento del conjunto S), y se considera el conjunto $\mathcal{A}$, como las **acciones** posibles, que son elementos asociados a cada estado en un tiempo dado. Además se usará la función $T : S \times \mathcal{A} \to S$ a aquella que recibe como parámetros un estado en un tiempo dado y una acción asociada a él en ese tiempo, y retorna el estado del siguiente tiempo, es decir, $s_{t+1} = T(s_t, a_t)$. Y por último, se considera la función de recompensa, que depende de un estado y una acción, es decir, $r(s, a)$.

Dado un factor de descuento $\gamma \in (0,1)$ y un estado inicial $s_0 \in S$, un problema de decisión a horizonte infinito toma la forma:

$$v(s_0) = \max_{a_t \in \mathcal{A}} \sum_{t=0}^{\infty} \gamma^t \cdot r(s_t, a_t)$$

con la restricciones:

$$s_t \in S, \quad s_{t+1} = T(s_t, a_t), \ \forall t \in [0, \infty)$$

Bellman introdujo el *Principio de Optimalidad* para descomponer el problema anterior:

Teorema 1.1 ***Principio de Optimalidad*** *Una táctica óptima tiene la propiedad de que cualquiera que sea el estado inicial y la decisión inicial, son las decisiones restantes las que deben constituir una táctica óptima en relación con el estado resultante de la primera decisión.*

Inspirados en el principio de optimalidad, se considera la decisión inicial aparte de las decisiones futuras, escribiendo el problema de decisión como:

$$\max_{a_0} \left\{ r(s_0, a_0) + \gamma \cdot \left[\max_{a_t} \sum_{t=1}^{\infty} \gamma^{t-1} \cdot r(s_t, a_t) : a_t \in \mathcal{A}, \ s_{t+1} = T(s_t, a_t), \ \forall t \geq 0 \right] \right\}$$

con las restricciones:

$$a_0 \in \mathcal{A}, \ s_1 = T(s_0, a_0)$$

Notemos que podemos simplificar la ecuación por medio de la **función valor**

$$V(s_0) = \max_{a_0} \Big\{ r(s_0, a_0) + \gamma \cdot V(s_1) \Big\}$$

con las restricciones:

$$a_0 \in \mathcal{A}, \ s_1 = T(s_0, a_0)$$

1.1.2. Ecuación de Bellman en un problema estocástico

Si se considera lo anterior referido a la ecuación de Bellman, pero para el caso en que la función T no es determinista, es decir, depende de una función de probabilidad $\mathbb{P}(s_{t+1}|s_t, a_t)$ que determina el estado siguiente s_{t+1}, a partir de s_t y a_t. Se tiene que la caracterización de la función de valor V quedaría como:

$$V(s_0) = \max_{a_0 \in \mathcal{A}} \left\{ r(s_0, a_0) + \gamma \cdot \sum_{s_1 \in S} \mathbb{P}(s_{t+1}|s_t, a_t) \cdot V(s_1) \right\}$$

Esta última caracterización recursiva de la función valor es la que nos permitirá calcular los V de manera iterativa, buscando su convergencia.

1.2. Método de Monte Carlo

Los métodos de Monte Carlo comprenden un conjunto de algoritmos no deterministas usado para aproximar mediante el empleo de muestras aleatorias expresiones matemáticas complejas y costosas de evaluar con exactitud. Los métodos de Monte Carlo son usados ampliamente en diversas áreas, por ejemplo, en problemas físicos que son particularmente eficientes para simular sistemas con muchos grados de libertad, tales como fluidos, materiales desordenados y estructuras celulares (ver [1, 7]).

Debe su nombre en referencia al Casino de Monte Carlo, que era considerada *la capital de los juego de azar* décadas atrás, es posible pensar en el juego de la ruleta como un generador simple de números aleatorios, y por otro lado, todo método de Monte Carlo debe estar construido a partir de un generador de números aleatorios que sea sencillo de implementar.

El método de Monte Carlo proporciona soluciones aproximadas a una gran variedad de problemas matemáticos, posibilitando la realización de experimentos con muestreos de números pseudoaleatorios en una computadora. El método es aplicable a cualquier tipo de problema estocástico o determinista.

Como ejemplo para ilustrar la potencia del método de Monte Carlo, se puede considerar el problema de integrar de manera aproximada una función continua $f : [0, 1] \to [0, 1]$, el método de Monte Carlo genera puntos aleatorios uniformemente distribuidos en el cuadrado $[0, 1]^2$, y con eso se define una variable aleatoria $X \sim$ Bernoulli(p) que es 1 si el punto cae bajo la curva y 0 si cae sobre ella, resultando que $\mathbb{E}(X) = \int_0^1 f(x)\,\mathrm{d}x$, y por la **Ley de los grandes números** es sabido que es posible estimar la esperanza por el promedio muestral de las v.a. $X_1, \ldots, X_n$ generadas:

$$\mathbb{E}(X) \approx \frac{1}{n} \sum_{\mathrm{i}=1}^{n} \frac{X_{\mathrm{i}}}{n}$$

Considerando como **n**, el número de muestras o datos, este método tiene un error absoluto de la estimación que decrece como $\frac{1}{\sqrt{n}}$, obtenido por el teorema central del límite.

Capítulo 2

Presentación del trabajo

El concepto de Aprendizaje Reforzado es uno de los principales paradigmas del Aprendizaje de Máquinas y busca establecer las *acciones* que un *agente* debe realizar en un *ambiente* para maximizar alguna *recompensa*. Su estudio es de amplia generalidad, encontrándose ejemplos en áreas tales como la teoría de juegos, la teoría de control, la investigación de operaciones, entre muchas otras (ver [9] y [14]).

El aprendizaje reforzado encuentra su base matemática en lo que se denominan Procesos de Decisión Markovianos (o MDPs), un tipo de proceso estocástico a tiempo discreto que permite modelar la interacción de un agente con un ambiente externo a través de un conjunto de reglas aleatorias. Los MDPs corresponden a una extensión de las cadenas de Markov y es posible establecer condiciones en las que un MDP reduce a una cadena Markov usual (ver [12], pág. 167).

2.1. Objetivo del trabajo

El objetivo principal de este trabajo corresponde al desarrollo de un algoritmo de estrategia óptima que servirá de base a la implementación de una herramienta tecnológica de aprendizaje adaptativo basada en la interacción entre un estudiante y un flujo de ejercicios de nivel de dificultad variable, y que busca maximizar el rendimiento promedio del estudiante.

Además, puesto que no se cuenta con datos reales de estudiantes, ni con una plataforma para obtener estos datos, es que se creó una función de distribución de probabilidad que permita simular un estudiante para realizar las pruebas y obtener los resultados que se mostrarán más adelante, pudiendo así probar el comportamiento del algoritmo realizado en el trabajo principal.

2.2. Definiciones Preliminares

En este capítulo se introducirán las definiciones matemáticas básicas que permitirán formalizar los conceptos de aprendizaje reforzado mediante MDPs. La mayoría de estas, son las definiciones estándar que se pueden encontrar en libros que toquen el tema de las MDPs (ver

[12]), y se incluyó la variante de considerar una variable de observación externa, cuya idea surgió de una tesis doctoral que explora algoritmos de RL y su interacción con humanos, para mejorar el aprendizaje educativo (ver [16]).

Definición 2.1 ***Conjunto de Estados/Acciones***
Consideraremos S y A dos conjuntos finitos. Un elemento $s \in S$ se denominara ***estado*** *y un elemento $a \in A$* ***acción****.*

Definición 2.2 ***Táctica***
Dados un conjunto de estados S, y un conjunto de acciones A, se denominara ***táctica*** *a una función $\pi : S \times \{0,1\} \times A \to [0,1]$, tal que $\forall s \in S,\ o \in \{0,1\}$ se tiene que $\sum_{a \in A} \pi(s,o,a) = 1$ (es decir, $\forall\ (s,o) \in S \times \{0,1\}$, $\pi(s,o,\cdot)$ es una distribución de probabilidad en A).*

Se conoce como ***táctica determinista*** *a aquellas tácticas que están asociadas a que todas las distribuciones de probabilidad asociadas sean puntuales. En este caso, para cada táctica determinista π construimos una nueva función: $\hat{\pi} : S \times \{0,1\} \to A$, donde $\hat{\pi}(s,o) = \hat{a}$, con $\hat{a}$ el único $a \in A$ tal que $\pi(s,o,a) = 1$.*

Definición 2.3 ***Función de recompensa***
Llamaremos ***función de recompensa*** *a una función $r : S \times \{0,1\} \times A \to \mathbb{R}$. También definimos el* ***conjunto de recompensas*** *como $R = \{r(s,o,a) \mid s \in S,\ o \in \{0,1\},\ a \in A\}$.*

Definición 2.4 ***Probabilidad de transición***
Dados $s \in S,\ o \in \{0,1\},\ a \in A$, consideremos una función $p_{s,o,a} : S \to [0,1]$, tal que $\sum_{s' \in S} p_{s,o,a}(s') = 1$. Es decir, $p_{s,o,a}$ es una función de probabilidad sobre S. Denotaremos por $\mathbb{P}(\ \cdot\ |s,o,a) := p_{s,o,a}(\cdot)$ a un elemento de la familia de ***probabilidades de transición****.*

Observación A lo largo de este trabajo se denotará por p a la familia de probabilidades de transición. Además se usará, siempre que no exista ambigüedad, la palabra *probabilidad* para referirse tanto a la función como al valor entre 0 y 1 que obtenemos a partir de una de estas funciones.

Definición 2.5 ***Agente de Decisiones con Recompensas y Observaciones***
Dados S, A, π, r, p, un ***Agente de Decisiones con Recompensas y Observaciones*** *(o* ***ADRO****), estará definido por los siguientes procesos estocásticos:*

- *X_t, con $t \in \mathbb{N}$ de variables aleatorias sobre S.*
- *O_t, con $t \in \mathbb{N}$ de variables aleatorias sobre $\{0,1\}$. Se le llamará* ***observación*** *a una realización de O_t.*
- *Y_t, con $t \in \mathbb{N}$ de variables aleatorias sobre A.*

- R_{t+1}, *con* $t \in \mathbb{N}$ *de variables aleatorias sobre* R.

Cuya distribución de probabilidades vendrá definida de manera inductiva sobre $t \in \mathbb{N}$ *por:*

- ◇ *Considerando como* $X_0 = s_0 \in S$, *y recibiendo una observación externa* O_0 *a partir de una distribución de Bernoulli. Luego se prosiguen los siguientes pasos, de manera iterativa.*
- ◇ *Estando en una realización de* X_t, *y una observación* O_t, *se obtiene una realización* Y_t *a partir de la distribución de probabilidad* $\pi(\ \cdot\ |\ X_t, O_t)$.
- ◇ *Luego se obtiene una realización* (X_{t+1}, R_{t+1}) *de la distribución de probabilidad* $\mathbb{P}(\cdot, \cdot | X_t, Y_t, O_t)$.
- ◇ *Entonces se obtiene una realización* O_{t+1} *a partir una distribución de probabilidad de Bernoulli de parámetro* λ *que depende de* X_{t+1} *(es decir,* $\lambda(X_{t+1})$ *o* $\lambda = F(X_{t+1})$, *con* F *una función por determinar en el capítulo de Simulaciones y Resultados).*

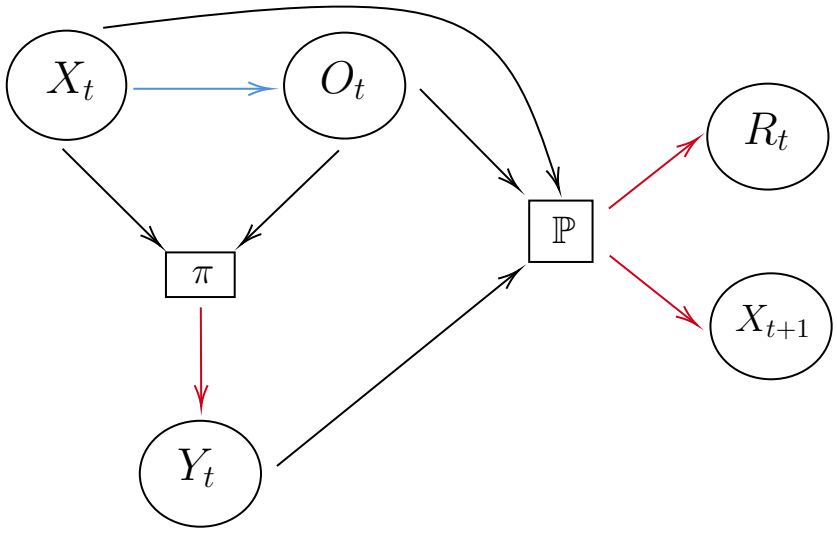

Figura 2.1: Diagrama de flujo

- *Los círculos indican las variables del proceso.*
- *Los rectángulos indican las funciones que aparecen en el proceso.*
- *La flecha azul indica la dependencia de una variable hacia otro proceso, que es exterior y por tanto, a priori no se controla.*
- *Las flechas rojas indican las variables output de las funciones.*
- *Las flechas negras indican las variables que reciben como parámetros o input las funciones.*

Definición 2.6 ***Recompensa acumulada***

Dado un ADRO, se denomina la ***recompensa acumulada*** *a la v.a.* $G_t = \sum_{\mathrm{i}=0}^{\infty} \gamma^{\mathrm{i}} \cdot R_{t+1+\mathrm{i}}$.

Donde $\gamma \in (0, 1)$ *se denomina* ***factor de descuento***.

Definición 2.7 ***Función de valor óptimo***

Se define v_π*, la* ***función de valor****, por:*

$$v_\pi(s) = \mathbb{E}_\pi(G_0 \mid X_0 = s),\ \forall s \in S$$

Y dado $s \in S$*, se denotará* v^**, la* ***función de valor óptimo*** *a:*

$$v^*(s) = \max_\pi v_\pi(s)$$

2.3. Intuiciones que motivan las definiciones

En el contexto del problema a desarrollar, se darán interpretaciones de las definiciones dadas previamente.

El **conjunto de estados** S (cf **Definición 2.1**) representa al conjunto de paquetes de ejercicios con el que se trabajarán, mediante una colección de enteros (Ej: $S = \{1, \ldots, 10\}$). Un elemento $s \in S$ representa un conjunto de ejercicios de nivel s, por lo que se considerará que al volver a un mismo nivel nunca se desarrollará exactamente el mismo ejercicio.

Por otro lado, el **conjunto de acciones** A corresponde al nivel que tiene más probabilidades de resultar ser el siguiente nivel de ejercicio en desplegarse.

La **observación** (cf **Definición 2.5**) da cuenta del rendimiento del estudiante en un determinado momento para cierto ejercicio, reflejándose con valor 0 si es incorrecto, y 1 si es correcto (la distribución de probabilidad que le da forma a este resultado, la cual anteriormente se presentó como la función F, es la que nos permite simular al estudiante). Esto permite interpretar la **táctica** (cf **Definición 2.2**) como la estrategia a seguir por el agente (máquina) para decidir por cierta acción, una vez que se está en un estado y se ha observado un rendimiento determinado.

Por otro lado, la **recompensa** busca representar el grado de aprendizaje de un alumno en un ejercicio y la **función de recompensa** corresponde a la distribución de probabilidad que determina el comportamiento del proceso estocástico, de que estando en un estado $s \in S$, habiéndose considerado una acción $a \in A$, y una observación o, se pase a un estado $s' \in S$, junto con una recompensa $r \in R$. Es decir, una vez que el estudiante ha contestado un ejercicio, y el sistema considera la acción a tomar, basándose en un componente azaroso conocido se determina el siguiente ejercicio en el flujo.

Teniendo en consideración el conjunto de estados S, el conjunto de acciones A, la táctica π, la función de recompensa r, y la probabilidad de transición p, el proceso estocástico (cf **Definición 2.5**) del modelo se define de manera inductiva en $t \in \mathbb{N}$:

- Estando en una realización de X_t, y una observación O_t, se utiliza la táctica $\pi(\ \cdot \mid X_t, O_t)$ para determinar una acción Y_t .
- Luego se obtiene la realización (X_{t+1}, R_{t+1}) de la probabilidad de transición $\mathbb{P}(s', r|X_t, Y_t, O_t)$.
- Entonces se espera una realización O_{t+1} de la variable aleatoria que vendría siendo la respuesta o rendimiento del alumno en el ejercicio X_{t+1}.

El **factor de descuento** es el que regulará la relevancia de las recompensas más cercanas en comparación a las futuras.

Finalmente, la **función de valor** es el valor que para cada estado se buscará maximizar, siendo la **función de valor óptimo** la solución del problema optimizado con su correspondiente π táctica óptima.

2.4. Modelamiento del problema

Dada la función de valor óptimo v^* definida en el capítulo 1, nos enfocaremos en encontrar una táctica π que maximice v_π, de esta manera podremos encontrar una táctica π^* que permitirá obtener v^*, construyendo valores cada vez más cercanos al óptimo deseado.

Para esto, recurriremos principalmente a dos métodos, los cuales son el **Método de Programación Dinámica** y el **Método de Monte Carlo**. Ambos métodos se enfocan en la manera en que se calculan parcialmente los v_π, a partir de una determinada táctica π. A esta fase se le llama **Evaluación**.

Luego, una vez calculado v_π, este se utiliza para actualizar la táctica, buscando una que mejore el resultado parcial que se tiene. A esta fase se le llama **Mejora**. Si efectivamente la táctica cambia, y mejora significativamente el resultado en comparación a la táctica anterior, se vuelve a realizar la primera fase, y así se itera.

El primer método, esto es Programación Dinámica, se enfoca en la resolución del problema de manera recursiva iterando valores parciales y no equilibrados (aproximados) de los v_π, para ir logrando mejores aproximaciones en cada iteración.

El segundo método, esto es Monte Carlo, se basa en realizar muchas simulaciones del problema a partir de valores aleatorios o con alto componente probabilístico, para así calcular los correspondientes v_π de la estrategia utilizada para dichas simulaciones.

Respecto a las ecuaciones utilizadas en el método de Programación Dinámica, la solución directa del sistema de ecuaciones es de alta complejidad computacional (ver [20]), por lo tanto para resolverla se prefiere un método iterativo.

A continuación presentaremos una revisión general de ambos métodos aplicados en contextos de optimización.

Capítulo 3

Trabajo Teórico Matemático

En esta sección, se profundizará en el desarrollo matemático y base teórica que sustenta los modelos a utilizar y por ende el presente trabajo.

3.1. Método de Programación Dinámica

En este método, dada una función V_0, se calcula V_1, mediante los valores de V_0. Luego de la misma forma que para V_1, se calcula V_2 (pero ahora utilizando V_1, en vez de V_0), y así sucesivamente, obteniendo una sucesión $(V_n)_n$, la cual converge a la función v_π, demostrando esto último en este capítulo. Para esto, primero veamos una propiedad que cumplen todas las funciones de valor v_π (desarrollo basado en [13] donde se realizan las demostraciones para el caso de la teoría estándar). Además, se usó como inspiración un contenido particular de una tesis (ver [16] p. 12) para incorporar las variables de observación en el proceso (idea secundaria de la tesis mencionada, pero principal para el presente trabajo respecto a lo que aporta directamente), y luego desarrollar la extensión a la teoría correspondiente.

3.1.1. Teoría estándar

En primer lugar se presentará el lema 3.1, que es un resultado parcial que después se utilizará para demostrar tanto el lema 3.2 y la proposición 3.4. Luego se pasará a replicar estos resultados, pero en el contexto de una extensión del problema.

Lema 3.1 *Dado un ADRO, y una función v_π definida por* $v_\pi(s) = \mathbb{E}_\pi\left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \,\middle|\, X_0 = s\right)$*, se cumple lo siguiente:*

$$v_\pi(s) = \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \left[r(s,o,\pi(s,o),j) + \gamma \cdot \mathbb{E}_\pi\left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \,\middle|\, X_1 = j\right)\right]$$

Demostración.

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left(\sum_{n=0}^{\infty}\gamma^n\cdot R_{n+1}\;\middle|\; X_0=s\right)\\
&= \sum_o p(o|s)\cdot\mathbb{E}_\pi\left(\sum_{n=0}^{\infty}\gamma^n\cdot R_{n+1}\;\middle|\; X_0=s,O_0=o\right)\\
&= \sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\mathbb{E}_\pi\left(\sum_{n=0}^{\infty}\gamma^n\cdot R_{n+1}\;\middle|\; X_0=s,O_0=o,X_1=j\right)\\
&= \sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\mathbb{E}_\pi\left(R_1+\sum_{n=1}^{\infty}\gamma^n\cdot R_{n+1}\;\middle|\; X_0=s,O_0=o,X_1=j\right)\\
&= \sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\\
&\qquad \mathbb{E}_\pi\left(r(X_0,O_0,\pi(X_0,O_0),X_1)+\sum_{n=1}^{\infty}\gamma^n\cdot R_{n+1}\;\middle|\; X_0=s,O_0=o,X_1=j\right)\\
&= \sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\Bigg[r(s,o,\pi(s,o),j)+\\
&\qquad \mathbb{E}_\pi\left(\sum_{n=1}^{\infty}\gamma^n\cdot R_{n+1}\;\middle|\; X_0=s,O_0=o,X_1=j\right)\Bigg]\\
&= \sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\left[r(s,o,\pi(s,o),j)+\mathbb{E}_\pi\left(\sum_{n=1}^{\infty}\gamma^n\cdot R_{n+1}\;\middle|\; X_1=j\right)\right]\\
&= \sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\left[r(s,o,\pi(s,o),j)+\mathbb{E}_\pi\left(\sum_{n=0}^{\infty}\gamma^{n+1}\cdot R_{n+2}\;\middle|\; X_1=j\right)\right]\\
&= \sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\left[r(s,o,\pi(s,o),j)+\gamma\cdot\mathbb{E}_\pi\left(\sum_{n=0}^{\infty}\gamma^n\cdot R_{n+2}\;\middle|\; X_1=j\right)\right]
\end{aligned}
$$

$$
\implies v_\pi(s)=\sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\left[r(s,o,\pi(s,o),j)+\gamma\cdot\mathbb{E}_\pi\left(\sum_{n=0}^{\infty}\gamma^n\cdot R_{n+2}\;\middle|\; X_1=j\right)\right]
$$

□

Continuando el desarrollo obtenido del lema 3.1, se demostrará el siguiente lema.

Lema 3.2 *Dado un ADRO, entonces v_π satisface:*

$$
v_\pi(s)=\sum_o p(o|s)\cdot\sum_j \mathbb{P}(j|s,o,\pi(s,o))\cdot\Big(r(s,o,\pi(s,o),j)+\gamma\cdot v_\pi(j)\Big)
$$

Demostración.

$$
\begin{aligned}
v_\pi(s) &= \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \left[r(s,o,\pi(s,o),j) + \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \;\middle|\; X_1 = j \right) \right] \\
&= \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \left[r(s,o,\pi(s,o),j) + \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \;\middle|\; X_0 = j \right) \right] \\
&= \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \Big[r(s,o,\pi(s,o),j) + \gamma \cdot v_\pi(j) \Big]
\end{aligned}
$$

$$
\implies v_\pi(s) = \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \Big(r(s,o,\pi(s,o),j) + \gamma \cdot v_\pi(j) \Big) \qquad \square
$$

Proposición 3.3 *Dado un ADRO y $0 < \gamma < 1$, se considera la función $T^\pi : \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$, tal que para $V \in \mathbb{R}^{|S|}$ y $s \in S$, se define como:*

$$
T^\pi(V)(s) = \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \Big(r(s,o,\pi(s,o),j) + \gamma \cdot V(j) \Big)
$$

Dado cualquier $V \in \mathbb{R}^{|S|}$, tenemos que $\lim_{n\to\infty} (T^\pi)^n(V) = v_\pi$.

A continuación el objetivo es demostrar que T^π es contractante en el espacio normado $(\mathbb{R}^{|S|}, ||\cdot||_\infty)$. Luego con esto se tendría que T^π tiene un único punto fijo, siendo este v_π, que es lo que se busca demostrar. Esto último se debe a que todo punto fijo de T^π satisface la ecuación del lema 3.2 (resulta directo verificar que v_π es punto fijo de T^π), y por el Teorema de punto fijo de Banach (ver [5]), existe un único $V^* \in \mathbb{R}^{|S|}$ tal que $T^\pi(V^*) = V^*$, el cual resulta ser el límite obtenido a partir de la sucesión $(T^\pi)^n(V)$, para un $V \in \mathbb{R}^{|S|}$ arbitrario (teniendo que T^π es contractante, se logra verificar que cada término de la sucesión cada vez está más cerca de V^*). Por lo anterior, se deduce que v_π es este único punto fijo.

Demostración. Sean $U, V \in \mathbb{R}^{|S|}$, entonces tenemos que:

$$
\begin{aligned}
\Big| T^\pi(V)(s) - T^\pi(U)(s) \Big| &= \Bigg| \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \Big(r(s,o,\pi(s,o),j) + \gamma \cdot V(j) \Big) - \\
&\qquad \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \Big(r(s,o,\pi(s,o),j) + \gamma \cdot U(j) \Big) \Bigg| \\
&= \left| \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \gamma \cdot (V(j) - U(j)) \right| \\
&\leq \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \gamma \cdot \Big| V(j) - U(j) \Big| \\
&\leq \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \gamma \cdot ||V - U||_\infty
\end{aligned}
$$

$$= \gamma \cdot ||V - U||_\infty \cdot \underbrace{\sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\cancel{\pi(s,o)})^{1}}_{1}$$

$$= \gamma \cdot ||V - U||_\infty < ||V - U||_\infty$$

$$\implies ||T^\pi(V) - T^\pi(U)||_\infty < ||V - U||_\infty$$

Con lo que se demuestra que la función T^π es contractante. □

Proposición 3.4 *Dado un ADRO y $0 < \gamma < 1$, sea v^* la función de valor óptimo definida más arriba. La función v^* satisface para cada $s \in S$ lo siguiente:*

$$v^*(s) = \sum_o p(o|s) \cdot \max_{a \in A} \left\{ \sum_{j \in S} \mathbb{P}(j|s,o,a) \cdot \Big(r(s,o,a,j) + \gamma \cdot v^*(j) \Big) \right\}$$

Demostración. Sea π una táctica determinista.

Luego, considerando $\pi = (a, \tilde{\pi})$ tenemos que:

$$\begin{aligned}
v^*(s) &= \max_\pi v_\pi(s) \text{ y usando el lema 3.1} \\
&= \max_\pi \left\{ \sum_o p(o|s) \cdot \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \left[r(s,o,\pi(s,o),j) + \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \;\middle|\; X_1 = j \right) \right] \right\} \\
&\overset{(1)}{=} \sum_o p(o|s) \cdot \max_\pi \left\{ \sum_j \mathbb{P}(j|s,o,\pi(s,o)) \cdot \left[r(s,o,\pi(s,o),j) + \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \;\middle|\; X_1 = j \right) \right] \right\} \\
&= \sum_o p(o|s) \cdot \max_{(a_0,\tilde{\pi})} \left\{ \sum_j \mathbb{P}(j|s,o,a_0) \cdot \left[r(s,o,a_0,j) + \gamma \cdot \mathbb{E}_{\tilde{\pi}} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \;\middle|\; X_1 = j \right) \right] \right\} \\
&= \sum_o p(o|s) \cdot \max_{(a_0,\tilde{\pi})} \left\{ \sum_j \mathbb{P}(j|s,o,a_0) \cdot \left[r(s,o,a_0,j) + \gamma \cdot \mathbb{E}_{\tilde{\pi}} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \;\middle|\; X_0 = j \right) \right] \right\} \\
&= \sum_o p(o|s) \cdot \max_{(a_0,\tilde{\pi})} \left\{ \sum_j \mathbb{P}(j|s,o,a_0) \cdot \Big[r(s,o,a_0,j) + \gamma \cdot v_{\tilde{\pi}}(j) \Big] \right\} \\
&= \sum_o p(o|s) \cdot \max_{a_0} \left\{ \max_{\tilde{\pi}} \left\{ \sum_j \mathbb{P}(j|s,o,a_0) \cdot \Big[r(s,o,a_0,j) + \gamma \cdot v_{\tilde{\pi}}(j) \Big] \right\} \right\} \\
&= \sum_o p(o|s) \cdot \max_{a_0} \left\{ \sum_j \mathbb{P}(j|s,o,a_0) \cdot \Big[r(s,o,a_0,j) + \gamma \cdot \max_{\tilde{\pi}} v_{\tilde{\pi}}(j) \Big] \right\} \\
&= \sum_o p(o|s) \cdot \max_{a_0} \left\{ \sum_j \mathbb{P}(j|s,o,a_0) \cdot \Big[r(s,o,a_0,j) + \gamma \cdot v^*(j) \Big] \right\}
\end{aligned}$$

$$\implies v^*(s) = \sum_o p(o|s) \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot v^*(j) \Big] \right\}$$

En donde en (1) se utilizó la independencia entre los valores de las acciones a considerar para la primera instancia. □

Proposición 3.5 (Teorema de punto fijo para MDP) *v^* es la única función que satisface el teorema anterior.*

Demostración. Sea $T : \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ tal que para $V \in R^{|S|}$ y $s \in S$ se define como:

$$T(V)(s) = \sum_o p(o|s) \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot V(j) \Big] \right\}$$

Análogamente a como se hizo antes con T^π, el objetivo es demostrar que T es contractante en el espacio normado $(\mathbb{R}^{|S|}, ||\cdot||_\infty)$. Con lo cual se tendría que T tiene único punto fijo, siendo este v^*, concluyendo así la demostración.

Sean $U, V \in \mathbb{R}^{|S|}$,

$$\begin{aligned}
|T(V)(s) - T(U)(s)| &= \left| \sum_o p(o|s) \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot V(j) \Big] \right\} - \right. \\
&\qquad \left. \sum_o p(o|s) \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot U(j) \Big] \right\} \right| \\
&= \left| \sum_o p(o|s) \cdot \left(\max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot V(j) \Big] \right\} \right. \right. \\
&\qquad \left. \left. - \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot U(j) \Big] \right\} \right) \right| \\
&\leq \sum_o p(o|s) \cdot \left| \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot V(j) \Big] \right\} \right. \\
&\qquad \left. - \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot \Big[r(s,o,a,j) + \gamma \cdot U(j) \Big] \right\} \right| \\
&\leq \sum_o p(o|s) \cdot \left| \gamma \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot (V(j) - U(j)) \right\} \right| \\
&\leq \sum_o p(o|s) \cdot \gamma \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot |V(j) - U(j)| \right\} \\
&\leq \gamma \cdot \sum_o p(o|s) \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \cdot ||V - U||_\infty \right\} \\
&\leq \gamma \cdot ||V - U||_\infty \cdot \underbrace{\sum_o p(o|s) \cdot \max_a \left\{ \sum_j \mathbb{P}(j|s,o,a) \right\}}_{1} 1
\end{aligned}$$

$$\leq \gamma \cdot ||V - U||_\infty < ||V - U||_\infty$$

$$\implies ||T(V) - T(U)||_\infty < ||V - U||_\infty$$

Teniendo así demostrado que T es contractante. □

3.1.2. Extensión de la teoría: táctica dependiente de 2 estados

Ahora buscaremos extender la teoría a trabajar, con tácticas dependientes de los últimos dos estados en vez de solo el último.

Como se vio anteriormente, se buscará demostrar la igualdad para la función de valor v_π pero que ahora viene dada por una táctica con dependencia de dos estados, para esto primero se demostrará el siguiente lema.

Lema 3.6 *Dado un ADRO, la función de valor v_π respectiva satisface lo siguiente:*

$$v_\pi(s) = \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Bigg(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg]$$

Demostración.

$$\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \;\middle|\; X_0 = s \right) \\
&= \sum_{o_0} p(o_0|s) \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \;\middle|\; X_0 = s, O_0 = o_0 \right) \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \;\middle|\; X_0 = s, O_0 = o_0, X_1 = s_1 \right) \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \mathbb{E}_\pi \left(R_1 + \sum_{n=1}^{\infty} \gamma^n \cdot R_{n+1} \;\middle|\; X_0 = s, O_0 = o_0, X_1 = s_1 \right)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \mathbb{E}_\pi \Bigg(r(X_0, O_0, \pi(X_0, O_0), X_1) + \\
&\qquad \sum_{n=1}^{\infty} \gamma^n \cdot R_{n+1} \;\Big|\; X_0 = s, O_0 = o_0, X_1 = s_1 \Bigg) \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \\
&\qquad \mathbb{E}_\pi \left(\sum_{n=1}^{\infty} \gamma^n \cdot R_{n+1} \;\Big|\; X_0 = s, O_0 = o_0, X_1 = s_1 \right) \Bigg] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \left[r(s, o_0, \pi(s, o_0), s_1) + \mathbb{E}_\pi \left(\sum_{n=1}^{\infty} \gamma^n \cdot R_{n+1} \;\Big|\; X_1 = s_1 \right) \right] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \left[r(s, o_0, \pi(s, o_0), s_1) + \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^{n+1} \cdot R_{n+2} \;\Big|\; X_1 = s_1 \right) \right] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \left[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \;\Big|\; X_1 = s_1 \right) \right] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \\
&\qquad \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \;\Big|\; X_1 = s_1, O_1 = o_1 \right) \Bigg] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\qquad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+2} \;\Big|\; X_1 = s_1, O_1 = o_1, X_2 = s_2 \right) \Bigg] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\qquad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \mathbb{E}_\pi \left(R_2 + \sum_{n=1}^{\infty} \gamma^n \cdot R_{n+2} \;\Big|\; X_1 = s_1, O_1 = o_1, X_2 = s_2 \right) \Bigg] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\qquad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \mathbb{E}_\pi \Bigg(r(X_1, O_1, \pi(s, o_0, X_1, O_1), X_2) + \\
&\qquad \sum_{n=1}^{\infty} \gamma^n \cdot R_{n+2} \;\Big|\; X_1 = s_1, O_1 = o_1, X_2 = s_2 \Bigg) \Bigg]
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Bigg(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \\
&\quad \mathbb{E}_\pi \left(\sum_{n=1}^{\infty} \gamma^n \cdot R_{n+2} \;\middle|\; X_1 = s_1, O_1 = o_1, X_2 = s_2 \right) \Bigg) \Bigg] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Bigg(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \\
&\quad \mathbb{E}_\pi \left(\sum_{n=1}^{\infty} \gamma^n \cdot R_{n+2} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Bigg(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \\
&\quad \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^{n+1} \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg] \\
&= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Bigg(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \\
&\quad \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg]
\end{aligned}
$$

□

Continuando con el desarrollo del lema 3.6, veremos el siguiente lema.

Lema 3.7 *Dado un ADRO, se tiene que v_π satisface:*

$$v_\pi(s) = \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}\Big(s_1|s, o_0, \pi(s, o_0)\Big) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \sum_{s_2} \mathbb{P}\Big(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)\Big) \cdot \Big(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \gamma \cdot v_\pi(s_2)\Big)\Bigg]$$

Demostración.

$$\begin{aligned} v_\pi(s) &= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0, s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0, s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\ &\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Bigg(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \\ &\quad \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \;\Big|\; X_0 = s_2\right)\Bigg)\Bigg] \\ &= \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\ &\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Big(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \gamma \cdot v_\pi(s_2)\Big)\Bigg] \end{aligned}$$

$$\implies v_\pi(s) = \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Big(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \gamma \cdot v_\pi(s_2)\Big)\Bigg]$$

□

Proposición 3.8 *Dado un ADRO y $0 < \gamma < 1$, se considera la función $T^\pi : \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$, que para $V \in \mathbb{R}^{|S|}$ y $s \in S$, se define como:*

$$T^\pi(V)(s) = \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Big(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \gamma \cdot V(s_2) \Big) \Bigg]$$

Dado cualquier $V \in \mathbb{R}^{|S|}$, tenemos que $\lim_{n\to\infty} (T^\pi)^n(V) = v_\pi$.

Demostración. Al igual que para la sección de la teoría estándar, el objetivo es demostrar que T^π es contractante en el espacio normado $(\mathbb{R}^{|S|}, ||\cdot||_\infty)$. Luego con esto se tendría que T^π tiene un único punto fijo, siendo v_π . Entonces veamos que T^π es contractante.

Sean $U, V \in \mathbb{R}^{|S|}$,

$$\begin{aligned}
\Big|T^\pi(V)(s) - T^\pi(U)(s)\Big| &= \Bigg| \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \\
&\quad \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \\
&\quad \Big(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \gamma \cdot V(s_2) \Big) \Bigg] - \\
&\quad \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \Bigg[r(s, o_0, \pi(s, o_0), s_1) + \\
&\quad \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \\
&\quad \Big(r(s_1, o_1, \pi(s, o_0, s_1, o_1), s_2) + \gamma \cdot U(s_2) \Big) \Bigg] \Bigg| \\
&= \Bigg| \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \gamma \cdot \Big(V(s_2) - U(s_2) \Big) \Bigg| \\
&\leq \gamma^2 \cdot \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot \Big| V(s_2) - U(s_2) \Big| \\
&\leq \gamma^2 \cdot \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s, o_0, s_1, o_1)) \cdot ||V - U||_\infty
\end{aligned}$$

$$
\begin{aligned}
&= \gamma^2 \cdot ||V - U||_\infty \cdot \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0)) \cdot \\
&\qquad \underbrace{\sum_{o_1} p(o_1|s_1) \cdot \underbrace{\sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi(s_1, o_1), s, o_0)}_{1}}_{1} \\
&= \gamma^2 \cdot ||V - U||_\infty \cdot \underbrace{\sum_{o_0} p(o_0|s) \cdot \underbrace{\sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi(s, o_0))}_{1}}_{1} \\
&= \gamma^2 \cdot ||V - U||_\infty < \ ||V - U||_\infty
\end{aligned}
$$

$\implies ||T^\pi(V) - T^\pi(U)||_\infty < ||V - U||_\infty$

Concluyendo así que T^π es contractante. □

Proposición 3.9 *Sea v^* la función de valor óptimo definida más arriba, con $0 < \gamma < 1$. La función v^* satisface para cada $s \in S$ lo siguiente:*

$$v^*(s) = \sum_{o_0} p(o_0|s) \cdot \max_{a_0} \left\{ \sum_{s_1} \mathbb{P}\Big(s_1|s,o_0,a_0\Big) \cdot \left[r(s,o_0,a_0,s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \max_{a_1} \left\{ \sum_{s_2} \mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big) \cdot \Big(r(s_1,o_1,a_1,s_2) + \gamma \cdot v^*(s_2)\Big) \right\} \right] \right\}$$

Demostración. Dado un ADRO, sea π una táctica determinista, de la forma $\pi = (\pi_0, \pi_1)$, donde π_0 corresponde a una táctica que solo depende del último estado (como la sección anterior), y π_1 es la táctica que depende de los dos últimos estados. Es decir, para esta parte π_0 solo se usará cuando se está comenzando el proceso.

Luego, considerando $\pi_1 = (a_1, \tilde{\pi})$ tenemos que:

$$\begin{aligned}
v^*(s) &= \max_{\pi} v_\pi(s) \text{ y usando el lema 3.6} \\
&= \max_{\pi} \left\{ \sum_{o_0} p(o_0|s) \cdot \sum_{s_1} \mathbb{P}(s_1|s,o_0,\pi(s,o_0)) \cdot \left[r(s,o_0,\pi(s,o_0),s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.\right. \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s,o_0,s_1,o_1,\pi(s,o_0,s_1,o_1)) \cdot \Bigg(r(s_1,o_1,\pi(s,o_0,s_1,o_1),s_2) + \\
&\quad \left.\left. \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \right] \right\} \\
&\overset{(1)}{=} \sum_{o_0} p(o_0|s) \cdot \max_{\pi} \left\{ \sum_{s_1} \mathbb{P}(s_1|s,o_0,\pi(s,o_0)) \cdot \left[r(s,o_0,\pi(s,o_0),s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.\right. \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s,o_0,s_1,o_1,\pi(s,o_0,s_1,o_1)) \cdot \Bigg(r(s_1,o_1,\pi(s,o_0,s_1,o_1),s_2) + \\
&\quad \left.\left. \gamma \cdot \mathbb{E}_\pi \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \right] \right\} \\
&= \sum_{o_0} p(o_0|s) \cdot \max_{(\pi_0,\pi_1)} \left\{ \sum_{s_1} \mathbb{P}(s_1|s,o_0,\pi_0) \cdot \left[r(s,o_0,\pi_0,s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.\right. \\
&\quad \sum_{s_2} \mathbb{P}(s_2|s,o_0,s_1,o_1,\pi_1(s,o_0,s_1,o_1),s,o_0) \cdot \Bigg(r(s_1,o_1,\pi_1(s,o_0,s_1,o_1),s_2) + \\
&\quad \left.\left. \gamma \cdot \mathbb{E}_{\pi_1} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \right] \right\} \\
&= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \left\{ \max_{\pi_1} \left\{ \sum_{s_1} \mathbb{P}(s_1|s,o_0,\pi_0) \cdot \left[r(s,o_0,\pi_0,s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.\right.\right.
\end{aligned}$$

$$
\begin{aligned}
&\sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi_1(s, o_0, s_1, o_1), s, o_0) \cdot \Bigg(r(s_1, o_1, \pi_1(s, o_0, s_1, o_1), s_2) + \\
&\gamma \cdot \mathbb{E}_{\pi_1} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg] \Bigg\} \Bigg\} \\
= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \Bigg\{ &\sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi_0) \cdot \Bigg[r(s, o_0, \pi_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\max_{\pi_1} \Bigg\{ \sum_{s_2} \mathbb{P}(s_2|s, o_0, s_1, o_1, \pi_1(s, o_0, s_1, o_1), s, o_0) \cdot \Bigg(r(s_1, o_1, \pi_1(s, o_0, s_1, o_1), s_2) + \\
&\gamma \cdot \mathbb{E}_{\pi_1} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg\} \Bigg] \Bigg\} \\
= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \Bigg\{ &\sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi_0) \cdot \Bigg[r(s, o_0, \pi_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\max_{(a_1, \tilde{\pi})} \Bigg\{ \sum_{s_2} \mathbb{P}(s_2|s, o_0, s, o_0, s_1, o_1, a_1) \cdot \Bigg(r(s_1, o_1, a_1, s_2) + \\
&\gamma \cdot \mathbb{E}_{\tilde{\pi}} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg\} \Bigg] \Bigg\} \\
= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \Bigg\{ &\sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi_0) \cdot \Bigg[r(s, o_0, \pi_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\max_{a_1} \Bigg\{ \max_{\tilde{\pi}} \Bigg\{ \sum_{s_2} \mathbb{P}(s_2|s, o_0, s, o_0, s_1, o_1, a_1) \cdot \Bigg(r(s_1, o_1, a_1, s_2) + \\
&\gamma \cdot \mathbb{E}_{\tilde{\pi}} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \Bigg) \Bigg\} \Bigg\} \Bigg] \Bigg\} \\
= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \Bigg\{ &\sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi_0) \cdot \Bigg[r(s, o_0, \pi_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\max_{a_1} \Bigg\{ \sum_{s_2} \mathbb{P}(s_2|s, o_0, s, o_0, s_1, o_1, a_1) \cdot \Bigg(r(s_1, o_1, a_1, s_2) + \\
&\gamma \cdot \max_{\tilde{\pi}} \left\{ \mathbb{E}_{\tilde{\pi}} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+3} \;\middle|\; X_2 = s_2 \right) \right\} \Bigg) \Bigg\} \Bigg] \Bigg\} \\
= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \Bigg\{ &\sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi_0) \cdot \Bigg[r(s, o_0, \pi_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \\
&\max_{a_1} \Bigg\{ \sum_{s_2} \mathbb{P}(s_2|s, o_0, s, o_0, s_1, o_1, a_1) \cdot \Bigg(r(s_1, o_1, a_1, s_2) + \\
\end{aligned}
$$

$$\gamma \cdot \max_{\tilde{\pi}} \left\{ \mathbb{E}_{\tilde{\pi}} \left(\sum_{n=0}^{\infty} \gamma^n \cdot R_{n+1} \;\middle|\; X_0 = s_0 \right) \right\} \Bigg) \Bigg\} \Bigg] \Bigg\}$$

$$= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \left\{ \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi_0) \cdot \Bigg[r(s, o_0, \pi_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.$$

$$\left. \max_{a_1} \left\{ \sum_{s_2} \mathbb{P}(s_2|s, o_0, s, o_0, s_1, o_1, a_1) \cdot \left(r(s_1, o_1, a_1, s_2) + \gamma \cdot \max_{\tilde{\pi}} v_{\tilde{\pi}}(s_2) \right) \right\} \Bigg] \right\}$$

$$= \sum_{o_0} p(o_0|s) \cdot \max_{\pi_0} \left\{ \sum_{s_1} \mathbb{P}(s_1|s, o_0, \pi_0) \cdot \Bigg[r(s, o_0, \pi_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.$$

$$\left. \max_{a_1} \left\{ \sum_{s_2} \mathbb{P}(s_2|s, o_0, s, o_0, s_1, o_1, a_1) \cdot \Big(r(s_1, o_1, a_1, s_2) + \gamma \cdot v^*(s_2) \Big) \right\} \Bigg] \right\}$$

□

Proposición 3.10 (Teorema de punto fijo para MDP con dependencia de 2 estados) *v^* es la única función que satisface la proposición anterior.*

Demostración. Dado un ADRO y $0 < \gamma < 1$. De manera parecida a antes, definimos la función $T : \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ para
$V \in R^{|S|}$ y $s \in S$ como:

$$T(V)(s) = \sum_{o_0} p(o_0|s) \cdot \max_{a_0} \left\{ \sum_{s_1} \mathbb{P}\Big(s_1|s, o_0, a_0\Big) \cdot \Bigg[r(s, o_0, a_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.$$

$$\left. \max_{a_1} \left\{ \sum_{s_2} \mathbb{P}\Big(s_2|s, o_0, s_1, o_1, a_1\Big) \cdot \Big(r(s_1, o_1, a_1, s_2) + \gamma \cdot V(s_2) \Big) \right\} \Bigg] \right\}$$

Luego como antes, se busca demostrar que T es contractante en el espacio normado $(\mathbb{R}^{|S|}, ||\cdot||_\infty)$, con esto se tendría que T tiene un único punto fijo, siendo este v^* .

$$\Big|T(V)(s) - T(U)(s)\Big| = \Bigg| \sum_{o_0} p(o_0|s) \cdot \max_{a_0} \left\{ \sum_{s_1} \mathbb{P}\Big(s_1|s, o_0, a_0\Big) \cdot \Bigg(r(s, o_0, a_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.$$

$$\left. \max_{a_1} \left\{ \sum_{s_2} \mathbb{P}\Big(s_2|s, o_0, s_1, o_1, a_1\Big) \cdot \Big(r(s_1, o_1, a_1, s_2) + \gamma \cdot V(s_2) \Big) \right\} \Bigg) \right\} -$$

$$\sum_{o_0} p(o_0|s) \cdot \max_{a_0} \left\{ \sum_{s_1} \mathbb{P}\Big(s_1|s, o_0, a_0\Big) \cdot \Bigg(r(s, o_0, a_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.$$

$$\left. \max_{a_1} \left\{ \sum_{s_2} \mathbb{P}\Big(s_2|s, o_0, s_1, o_1, a_1\Big) \cdot \Big(r(s_1, o_1, a_1, s_2) + \gamma \cdot U(s_2) \Big) \right\} \Bigg) \right\} \Bigg|$$

$$\leq \sum_{o_0} p(o_0|s) \cdot \Bigg| \max_{a_0} \left\{ \sum_{s_1} \mathbb{P}\Big(s_1|s, o_0, a_0\Big) \cdot \Bigg(r(s, o_0, a_0, s_1) + \gamma \cdot \sum_{o_1} p(o_1|s_1) \cdot \right.$$

$$
\begin{aligned}
&\qquad\qquad \max_{a_1}\left\{\sum_{s_2}\mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big)\cdot\Big(r(s_1,o_1,a_1,s_2)+\gamma\cdot V(s_2)\Big)\right\}\Bigg)\Bigg\}- \\
&\max_{a_0}\left\{\sum_{s_1}\mathbb{P}\Big(s_1|s,o_0,a_0\Big)\cdot\Bigg(r(s,o_0,a_0,s_1)+\gamma\cdot\sum_{o_1}p(o_1|s_1)\cdot \right. \\
&\qquad\qquad \left.\max_{a_1}\left\{\sum_{s_2}\mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big)\cdot\Big(r(s_1,o_1,a_1,s_2)+\gamma\cdot U(s_2)\Big)\right\}\Bigg)\right\}\Bigg| \\
&\leq \sum_{o_0}p(o_0|s)\cdot\max_{a_0}\left\{\sum_{s_1}\mathbb{P}\Big(s_1|s,o_0,a_0\Big)\cdot\gamma\cdot\sum_{o_1}p(o_1|s_1)\cdot\right. \\
&\qquad\left|\max_{a_1}\left\{\sum_{s_2}\mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big)\cdot\Big(r(s_1,o_1,a_1,s_2)+\gamma\cdot V(s_2)\Big)\right\}-\right. \\
&\qquad\left.\left.\max_{a_1}\left\{\sum_{s_2}\mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big)\cdot\Big(r(s_1,o_1,a_1,s_2)+\gamma\cdot U(s_2)\Big)\right\}\right|\quad\right\} \\
&\leq \sum_{o_0}p(o_0|s)\cdot\max_{a_0}\left\{\sum_{s_1}\mathbb{P}\Big(s_1|s,o_0,a_0\Big)\cdot\gamma\cdot\sum_{o_1}p(o_1|s_1)\cdot\right. \\
&\qquad\left.\max_{a_1}\left\{\sum_{s_2}\mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big)\cdot\gamma\cdot\Big|V(s_2)-U(s_2)\Big|\right\}\right\} \\
&\leq \sum_{o_0}p(o_0|s)\cdot\max_{a_0}\left\{\sum_{s_1}\mathbb{P}\Big(s_1|s,o_0,a_0\Big)\cdot\gamma\cdot\sum_{o_1}p(o_1|s_1)\cdot\right. \\
&\qquad\left.\max_{a_1}\left\{\sum_{s_2}\mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big)\cdot\gamma\cdot\Big|\Big|V-U\Big|\Big|_\infty\right\}\right\} \\
&= \gamma\cdot\Big|\Big|V-U\Big|\Big|_\infty\cdot\sum_{o_0}p(o_0|s)\cdot\max_{a_0}\left\{\sum_{s_1}\mathbb{P}\Big(s_1|s,o_0,a_0\Big)\cdot\gamma\cdot\sum_{o_1}p(o_1|s_1)\cdot\right. \\
&\qquad\left.\underbrace{\max_{a_1}\left\{\underbrace{\sum_{s_2}\mathbb{P}\Big(s_2|s,o_0,s_1,o_1,a_1\Big)}_{1}\right\}}_{1}\right\} \\
&= \gamma^2\cdot\Big|\Big|V-U\Big|\Big|_\infty\cdot\underbrace{\sum_{o_0}p(o_0|s)\cdot\max_{a_0}\left\{\underbrace{\sum_{s_1}\mathbb{P}\Big(s_1|s,o_0,a_0\Big)\cdot\underbrace{\sum_{o_1}p(o_1|s_1)}_{1}}_{1}\right\}}_{1} \\
&= \gamma^2\cdot\Big|\Big|V-U\Big|\Big|_\infty < \Big|\Big|V-U\Big|\Big|_\infty \\
\implies &\Big|\Big|T(V)-T(U)\Big|\Big|_\infty < \Big|\Big|V-U\Big|\Big|_\infty
\end{aligned}
$$

Teniendo finalmente que T es contractante. □

3.2. Método de Monte Carlo

Para una táctica π fija, se considera el ADRO correspondiente para generar el proceso estocástico asociado $H = (\{X_n\}, \{O_n\}, \{A_n\}, \{R_n\})$, donde X_n son los estados visitados en los tiempos correspondientes; $\{O_n\}$ sus respectivas observaciones; $\{A_n\}$ las acciones asociadas, tomadas a partir de la táctica considerada al comienzo; y $\{R_n\}$ las recompensas obtenidas en cada tiempo.

Luego, para $s \in S,\ t \in \mathbb{N}$, definimos:

- $N^t(s)$ como la cantidad de veces que se visitó s, considerando hasta el tiempo t.
- $S^t(s)$ corresponde a la suma total de retornos desde s, considerando hasta el tiempo t.
- $V^t(s) = \frac{S^t(s)}{N^t(s)}$ corresponde al promedio de los retornos desde s, considerando hasta el tiempo t.

Reescribiendo lo anterior de manera adecuada, tenemos que:

- $N^t(s) = \sum_{\mathrm{i}=0}^{t} \mathbf{1}_{X_\mathrm{i}=s}$
- $S^t(s) = \sum_{\mathrm{i}=0}^{t} \mathbf{1}_{X_\mathrm{i}=s} \cdot G_\mathrm{i}$
- $V^t(s) = \dfrac{\sum_{\mathrm{i}=0}^{t} \mathbf{1}_{X_\mathrm{i}=s} \cdot G_\mathrm{i}}{\sum_{\mathrm{i}=0}^{t} \mathbf{1}_{X_\mathrm{i}=s}}$

Suponiendo que no existen estados transitorios, y por una adecuada versión de la Ley de los Grandes Números (LDGN), tenemos que $V^t(s) \longrightarrow v_\pi(s)$, cuando $N^t(s) \longrightarrow +\infty$.

En la literatura usualmente se pueden ver esbozos de demostraciones referidas a la justificación teórica de versiones más simples del método, sin precisar en las variables y considerando la LDGN estándar donde las variables aleatorias son iid, sin embargo, en los modelos de Reinforcement Learning estas variables rara vez son iid (en la pág 4 de [20], ver los ejemplos introductorios), por lo que es necesario precisar más en el análisis sobre las hipótesis, adaptado al caso particular del contexto del problema abordado, y como utilizar esto buscando una variante de la LDGN.

Sea $s \in S,\ t \in \mathbb{N}$ fijos, y considerando $k = N^t(s),\ x_j = G_{n(j)},$ con $j \in \{1, \ldots, k\}$, donde $n(j)$ es tal que $X_{n(j)} = s$. Entonces se define:

$$\begin{aligned}
\mu_k &:= V^{n(k)}(s) \\
&= \frac{1}{k} \cdot \sum_{j=1}^{k} x_j \\
&= \frac{1}{k} \cdot \left(x_k + \sum_{j=1}^{k-1} x_j \right) \\
&= \frac{1}{k} \cdot (x_k + (k-1) \cdot \mu_{k-1}) \\
&= \mu_{k-1} + \frac{1}{k} \cdot (x_k - \mu_{k-1})
\end{aligned}$$

De esta manera, se puede establecer una forma iterativa de ir calculando los promedios de a poco, de un paso a otro:

$$V^{n(k)}(s) = V^{n(k-1)}(s) + \frac{1}{N^{n(k)}(s)} \cdot \left(G_{n(k)} - V^{n(k-1)}(s) \right)$$

Esto es útil para no tener que guardar todos los valores y luego calcular los promedios, debido a que esa forma no es eficiente en el uso de la memoria. Por lo que se opta por un cálculo iterativo de los promedios.

Capítulo 4

Simulaciones y resultados

En esta sección se realizarán simulaciones hechas en un notebook personal con las siguientes características:

- Procesador: Intel(R) Core(TM) i5-6200U CPU @ 2.3GHz (4 CPUs), 2.4GHz
- Memoria RAM: 3.88 GB utilizable

A partir de ciertos parámetros iniciales y aplicando los métodos mencionados en el capítulo anterior, se busca obtener la táctica π^*, mostrando los resultados correspondientes, considerando primero la táctica dependiente del estado anterior, y luego considerando a la extensión de la táctica dependiente de los dos estados previos. Esto contemplado tanto para el método de Programación Dinámica, como para el método de Monte Carlo.

4.1. Parámetros

Consideraremos los siguientes parámetros base (para determinados resultados serán cambiados, mencionando por cuales):

- $\varepsilon_E = 10^{-3}$ (**Error relativo de V en la Evaluación**), se utiliza solo en PD.
- $\varepsilon_M = \begin{cases} 0 & \text{para PD} \\ 10^{-3} & \text{para MC} \end{cases}$ (**Error relativo de V en la Mejora**)
- $\gamma = 0{,}8$ (**Factor de descuento**), también conocido como pérdida de memoria, que establece el peso con que influye la información más actualizada respecto a la antigua. Se eligió este parámetro para mantener una consideración importante de los ejercicios vistos.
- $|S| = N = 30$, al considerarse como un número aceptable de variedad en niveles de ejercicios para considerar que formen parte de una misma ronda.
- $\text{media}_{est} = 0{,}6$ (**Probabilidad de acierto base de un estudiante al contestar el**

ejercicio de nivel 1), indica que al contestar un ejercicio de nivel 1, es levemente más probable contestarlo correctamente que incorrectamente.

- Ronda = 200 (se utiliza solo en MC, se refiere a la cantidad de ejercicios consecutivos en la simulación), en principio utilizado para hacer las simulaciones iniciales sin consumir grandes recursos del computador, pero el ideal es que sea un número grande, para así conseguir un equilibrio en la convergencia.

- En las rondas, se comenzará desde el nivel 1.

Además se utilizaron las siguientes funciones de recompensa y probabilidad de transición:

Dado $s \in S,\ a \in A,\ o \in \{0,1\}$, la **recompensa** $r(s,a,o)$ viene dada por:

$r(s,a,o)$	$s \leq a$	$a < s$
$o = 0$	-1	0

$r(s,a,o)$	$s < a$	$a \leq s$
$o = 1$	2	1

Tabla 4.1: Función de recompensa

Los valores de la recompensa fueron seleccionados pensando en que si un ejercicio es contestado de manera incorrecta, se incentive negativamente el seguir adelante (como haciendo la analogía de primero buscar reforzar los ejercicios previos), y si es contestado correctamente, se quiere incentivar el avanzar.

Dado $s_t \in S,\ a_t \in A,\ o_t \in \{0,1\}$, la **probabilidad** $\mathbb{P}(s_{t+1}|s_t,a_t,o_t)$ viene dada por:

$\mathbb{P}(s_{t+1}\|s_t,o_t,a_t)$	$s_{t+1} = a_t + 1$	$s_{t+1} = a_t$	$s_{t+1} = a_t - 1$
$a_t = 1$	25 %	75 %	0 %
$a_t = N$	0 %	75 %	25 %
$1 < a_t < N$	25 %	50 %	25 %

Tabla 4.2: Función de probabilidad a 1 estado

Con estos valores se busca que sea más probable que el siguiente estado corresponda al nivel de la acción elegida, pero dejando una probabilidad pequeña a la opción de que pueda ser un ejercicio de nivel vecino.

A continuación se expone las probabilidades consideradas para el caso de la extensión con 2 estados de dependencia, teniendo que las recompensas siguen dependientes de solo el estado anterior, por lo que siguen siendo igual que antes. Teniendo así que la probabilidad de transición para s_{t+1}, dado $s_t, o_t, a_t, s_{t-1}, o_{t-1}$ (donde a_t es la acción correspondiente obtenida de $\pi(s_t,o_t)$ y s_{t-1} es el estado previo a s_t, con o_{t-1} su correspondiente observación) de:

$\mathbb{P}\big(s_{t+1}\big\vert s_t, o_t, a_t, s_{t-1}, o_{t-1}\big)$		$o_t = 1$		$o_t = 0$	
		$o_{t-1} = 1$	$o_{t-1} = 0$	$o_{t-1} = 1$	$o_{t-1} = 0$
$a_t = s_t + 1$	$s_{t+1} = a_t + 1$	10 %	0 %	10 %	0 %
	$s_{t+1} = a_t$	60 %	60 %	60 %	40 %
	$s_{t+1} = a_t - 1$	30 %	40 %	30 %	60 %
$a_t = s_t$	$s_{t+1} = a_t + 1$	40 %	10 %	40 %	0 %
	$s_{t+1} = a_t$	50 %	80 %	50 %	80 %
	$s_{t+1} = a_t - 1$	10 %	10 %	10 %	20 %
$a_t = s_t - 1$	$s_{t+1} = a_t + 1$	80 %	60 %	80 %	10 %
	$s_{t+1} = a_t$	20 %	30 %	20 %	80 %
	$s_{t+1} = a_t - 1$	0 %	10 %	0 %	10 %

Tabla 4.3: Función de probabilidad a 2 estados

Para evitar oscilaciones en los niveles altos, y así simular un término anticipado de la ronda de ejercicios, se puso un tope de bajada para las probabilidades mostradas anteriormente (tanto dependencia para un estado previo, como para dos estados previos) de los casos en que $s_t \geq N - 1$, mostrándose en la siguiente tabla:

$\mathbb{P}(s_{t+1}\vert o_t)$	$s_{t+1} = N - 1$	$s_{t+1} = N$
$o_t = 0$	30 %	70 %
$o_t = 1$	0 %	100 %

Tabla 4.4: Probabilidad de tope de bajada

Además, para iniciar el algoritmo, necesitamos entregarle una táctica inicial. Consideraremos la siguiente táctica, la cual no es mayormente especial más allá del hecho de ser factible, la definiremos como:

$$\text{Dado } s \in S, \quad \pi(s, \cdot) = \begin{cases} s + 1 & \text{si} \quad s < N \\ \\ s & \text{si} \quad s = N \end{cases}$$

4.2. Representación de las tácticas

Una táctica se representará a partir de un gráfico como el que se observa continuación, donde el eje X indica el nivel s del ejercicio que recibe como parámetro la función π, el eje Y representa la acción asociada a esta función para este punto, el color rojo en el gráfico, indica que asociado a ese ejercicio hay una respuesta incorrecta; y azul, indica una respuesta correcta.

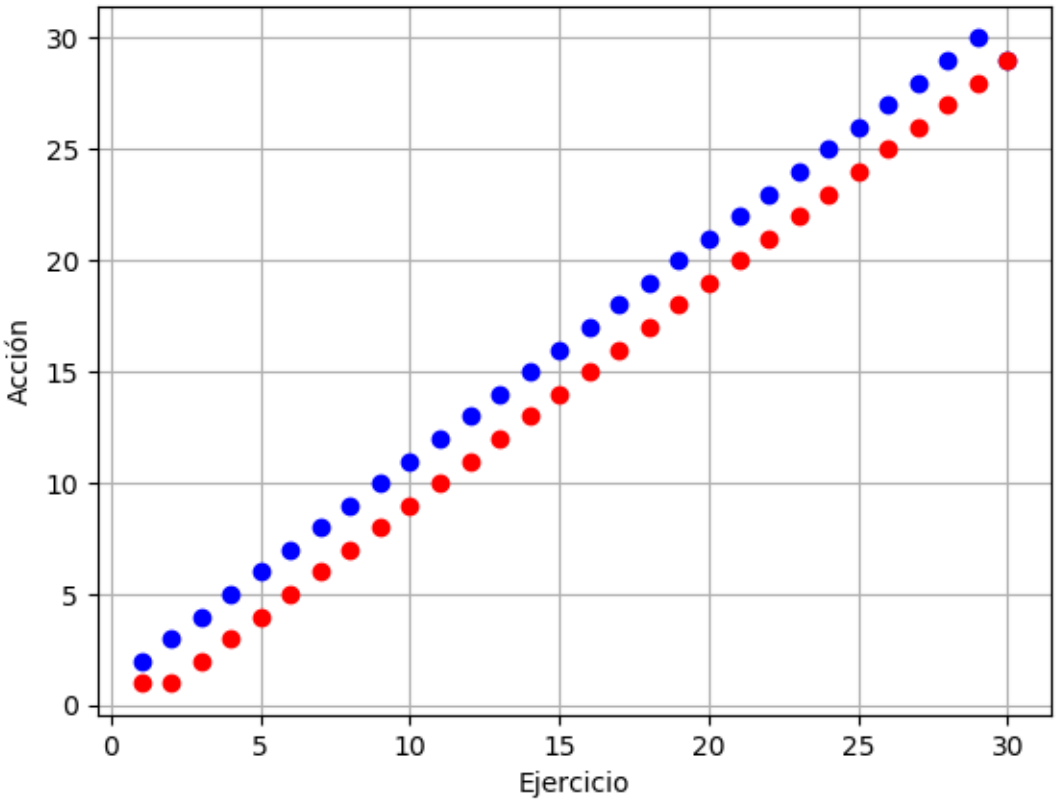

Figura 4.1: Gráfico de táctica, mostrando la acción elegida por nivel de ejercicio.

El ejemplo de táctica anterior coincide con el que se verá después como una estrategia óptima, y utilizaremos esta táctica como base para realizar las pruebas de la función que tiene como fin simular de forma básica a un estudiante.

Además, para graficar la táctica correspondiente a la dependencia de los dos estados previos, primero se determinó si se mostrará la táctica para $o_{t-1} = 0$ o $o_{t-1} = 1$. Luego consideramos que la táctica asociada a $o_t = 0$ se mostrará en color rojo, y la correspondiente a $o_t = 1$ se mostrará en color azul. Para representar el valor de s_{t-1} se utilizará la siguiente simbología, y así lograr condensar la información de los parámetros que recibe la función de táctica en un solo gráfico, logrando facilitar la visualización y más adelante el posterior análisis de los resultados.

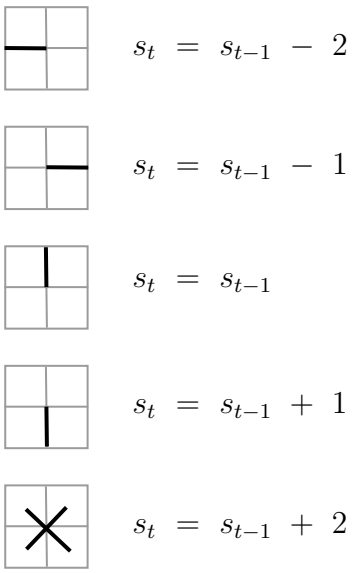

Figura 4.2: Simbología de gráficos de tácticas a 2 estados

Donde el eje X representa el valor de s_t, y el eje Y representa la acción correspondiente a la táctica, obteniendo un gráfico como el siguiente:

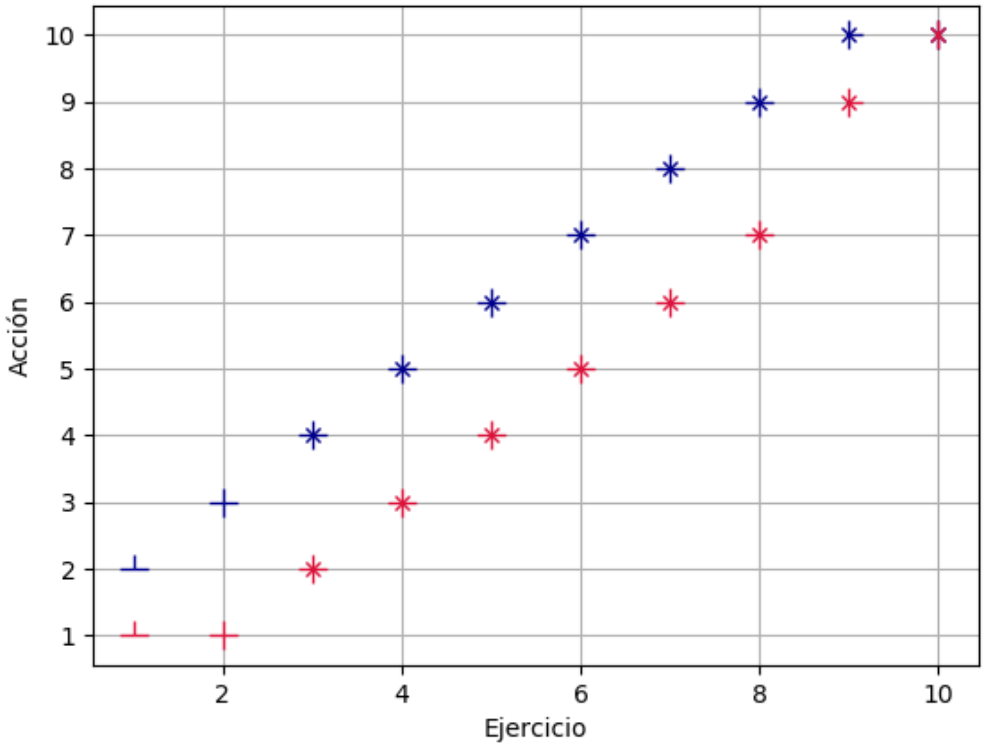

Figura 4.3: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

4.3. Simulador estudiante

Como se mencionó anteriormente en la presentación del trabajo, puesto que no se cuentan con datos reales, ni con una plataforma para obtener estos datos, fue necesario implementar una función que simule de manera básica un estudiante. Esta función, que la consideraremos como la probabilidad de que se conteste de manera correcta un determinado nivel de ejercicio, buscamos que satisfaga como mínimo las siguientes propiedades:

1. Probabilidad no puede llegar ni a 0, ni a 1, más precisamente, para $\varepsilon > 0$ pequeño, se busca que la probabilidad esté en el intervalo $(\varepsilon, 1-\varepsilon)$. Así la fórmula evita otorgar resultados que sean solo respuestas correctas o incorrectas (no se consideran los extremos). Para hacer las pruebas y posteriormente simulaciones, se fija $\varepsilon = 0{,}05$.

2. A mejor rendimiento, se tenga una mayor probabilidad.

3. Se tenga menor probabilidad a mayor nivel, manteniendo el resto de los parámetros iguales.

4. A mayor cantidad de ejercicios resueltos, aumente el cómo afecta el historial en la función.

5. Mientras más antiguas las respuestas, se consideran menos que las más recientes, así el factor de descuento γ o de memoria, juega un rol importante para la función.

Por lo enumerado anteriormente, consideramos un simulador de estudiante dado por la fórmula:

$$probObs = \text{media}_{est} + \frac{0{,}95 - \text{media}_{est}}{\text{rend} + (1 - \text{rend}) \cdot s} \cdot \text{mín}\left\{1, \frac{k}{N}\right\}$$

Donde:

- probObs = la probabilidad de que un estudiante conteste positivamente el ejercicio preguntado en un momento dado
- media_{est} = base de probabilidad con que comienza el estudiante
- k = cantidad de ejercicio hechos de la ronda actual hasta el momento
- s = nivel del ejercicio actual
- N = nivel máximo a considerar de los ejercicios
- $\text{rend} = \begin{cases} \dfrac{1-\gamma}{1-\gamma^k} \cdot \displaystyle\sum_{\text{i}=1}^{k} \gamma^{\text{i}-1} \cdot o_{\text{i}} & si \quad k > 0 \\ 0 & si \quad k = 0 \end{cases}$

Un ejemplo de simulación de un estudiante se representa en el siguiente flujo, donde en ambos gráficos el eje X represente el momento temporal de la trayectoria del estudiante contestando la ronda de ejercicios. En el primer gráfico, el eje Y representa el nivel del ejercicio visto en el momento correspondiente, además los círculos rojos se refieren a que el estudiante contestó de manera incorrecta el ejercicio, y azul si es que fue una respuesta correcta. En el segundo gráfico, el eje Y representa las probabilidades de acierto (o de contestar de manera correcta el respectivo ejercicio) asociadas al estudiante en aquel momento. Donde se utilizaron los parámetros antes fijados, pero considerando $Ronda = 60$.

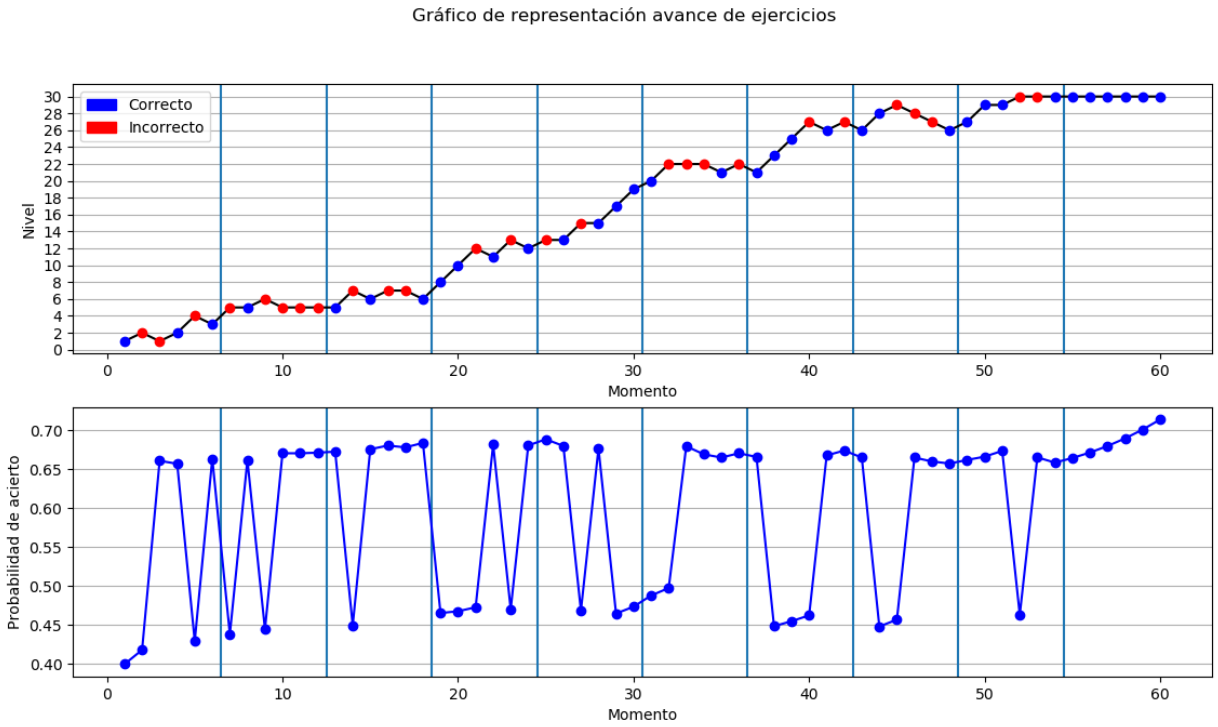

Figura 4.4: Usando los parámetros $N = 30$, Ronda $= 60$, $\text{media}_{est} = 0{,}6$

Luego, generando 1 000 000 simulaciones con los mismos parámetros de la anterior, extrayendo y guardando los datos, tenemos que los promedios de las probabilidades asociadas por nivel se ven reflejadas en el siguiente gráfico. Donde en el eje X se posicionan los niveles de los ejercicios que consideramos, y en el eje Y el promedio de las probabilidades.

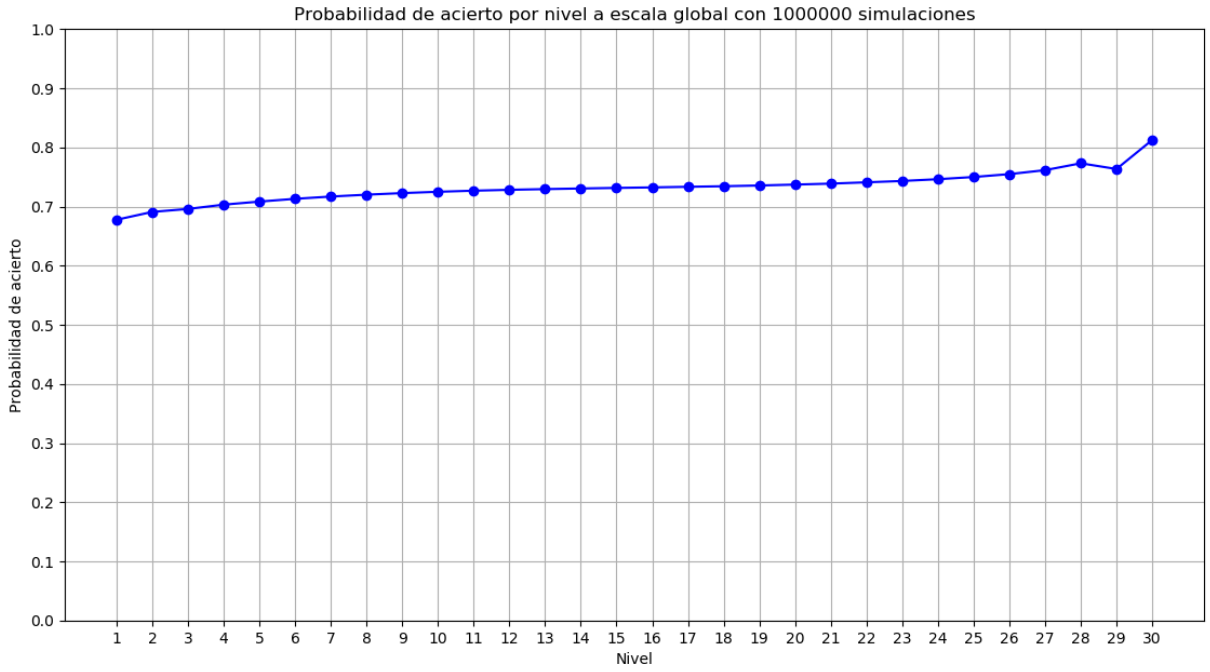

Figura 4.5: Usando los parámetros $N = 30$, $Ronda = 60$, $\text{media}_{est} = 0{,}6$

Si analizamos para determinados momentos la distribución de probabilidades por estado, tenemos el siguiente gráfico. Donde el eje X representa los niveles de los ejercicios, y el eje Y representa los promedios de las probabilidades de acierto.

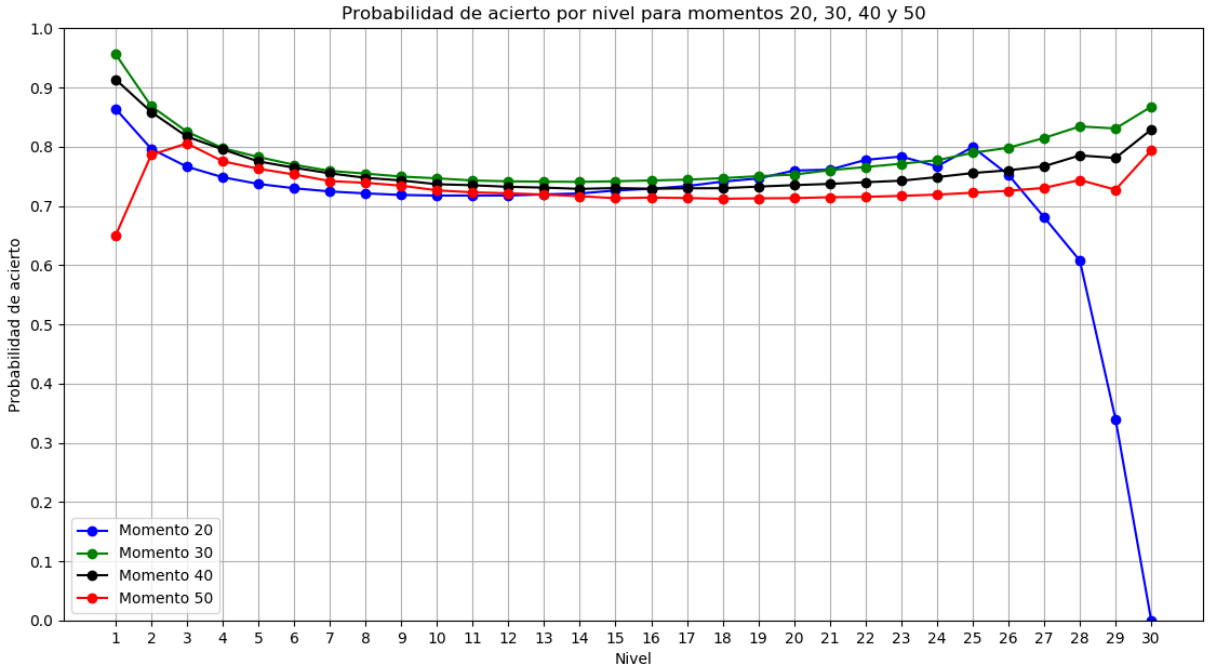

Figura 4.6: Gráfico de probabilidad media de acierto respecto al nivel de ejercicio.

4.4. Resultados generales

En esta sección se presentarán los resultados obtenidos producto de aplicar en forma paralela los métodos de Programación Dinámica y de Monte Carlo para obtener las tácticas respectivas. Para esto se utilizaron los parámetros mencionados en la sección de más arriba, y la función que simula un estudiante.

Presentaremos para cada resultado (táctica π) obtenido, su tiempo de ejecución, iteraciones realizadas (se puede ver como la cantidad de veces que se calculó el valor V), y el respectivo valor V, que recordemos que es la recompensa acumulada esperada asociada al proceso estocástico usando la táctica π.

Primero se mostrarán los resultados obtenidos para programación dinámica, más precisamente para el caso en que las tácticas dependen solo del estado actual, y luego se verá para el caso en que las tácticas dependen del estado actual y el estado previo. Luego se mostrarán los resultados respecto al método de Monte Carlo, considerando el mismo orden de casos en las tácticas que para Programación Dinámica. Recordemos que la diferencia de ambos métodos, solo radica en la manera de calcular los valores de v_π asociados a una táctica π.

La representación de los gráficos sigue el mismo esquema o significancia en su notación que lo mencionado en la sección anterior.

4.4.1. Programación Dinámica

Este método efectúa inicialmente todos los cálculos teóricos para determinar la táctica óptima con valores fijos iniciales que provienen de la variable aleatoria, por lo que es un método determinista, ya que no presenta variación si no se modifican los parámetros, solo el tiempo de ejecución puede variar ligeramente debido a la carga propia del computador que no tiene relación con el proceso del algoritmo, por lo que este valor se colocará como un valor aproximado.

Como ya se mencionó en una sección pasada, el algoritmo funciona con 2 fases que son de distinta naturaleza. En la primera fase, es donde se aplica el método de Programación Dinámica, para calcular iterativamente v_π asociado a una estrategia π formando así sub-iteraciones en el algoritmo; la segunda fase es general, en donde se recalcula la táctica que maximice el retorno acumulado esperado, entonces las iteraciones generales vienen siendo la cantidad de veces que se calcula la táctica. Las sub-iteraciones se representarán con números separados con guiones, para dejar en claro que pertenecen a iteraciones generales distintas (Ej: *2-3*, indica que en la primera iteración general, hubo 2 sub-iteraciones; y que en la segunda iteración general, hubo 3).

Táctica dependiente del estado previo

Prueba 1:

- Nivel = 30
- Tiempo: $\sim$ 0.26s
- Iteraciones: 29-9-9 (3 generales)
- Valor V: 5.8

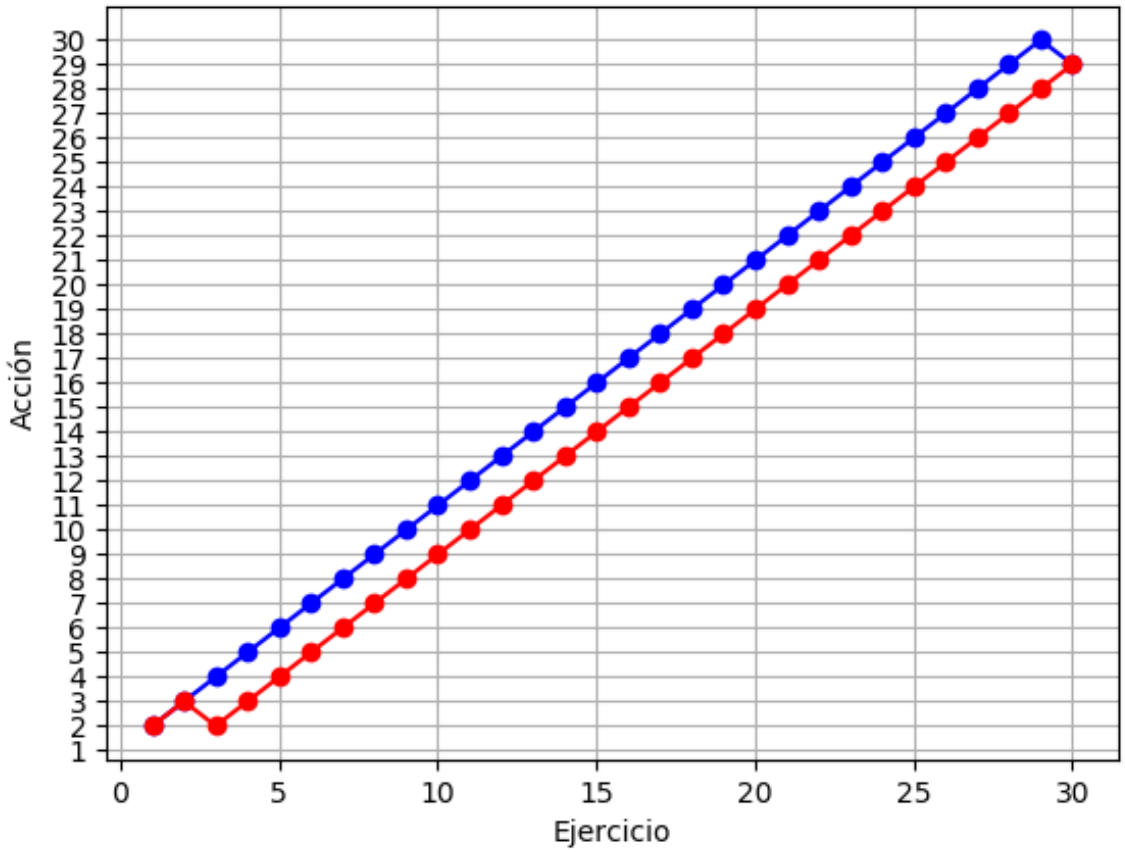

Figura 4.7: Gráfico de táctica, mostrando la acción elegida por nivel de ejercicio.

Táctica dependiente de los dos estados previos

Para el algoritmo en que se trabajó con tácticas que reciben como parámetros los dos últimos estados, también se consideró que la familia de funciones de probabilidad reciben adicionalmente como parámetro el estado previo. Por lo que, debido a que la estructura del algoritmo es muy parecida pero con algunas diferencias, primero se evaluó utilizando la familia de probabilidades utilizadas para el caso anterior (dependencia un estado), es decir, se restringió el algoritmo en su dependencia de las probabilidades a un estado (como fingiendo que solo le paso los datos del estado previo). Esto permite asegurarse que el modelo extendido a dos estados está bien definido, buscando corroborar empíricamente una compatibilidad al obtener los mismos resultados que para el caso de un estado.

Prueba 2:

- Nivel = 30
- Tiempo: $\sim$ 0.17s
- Iteraciones: 2-2-2 (3 generales)
- Valor V: 4.26

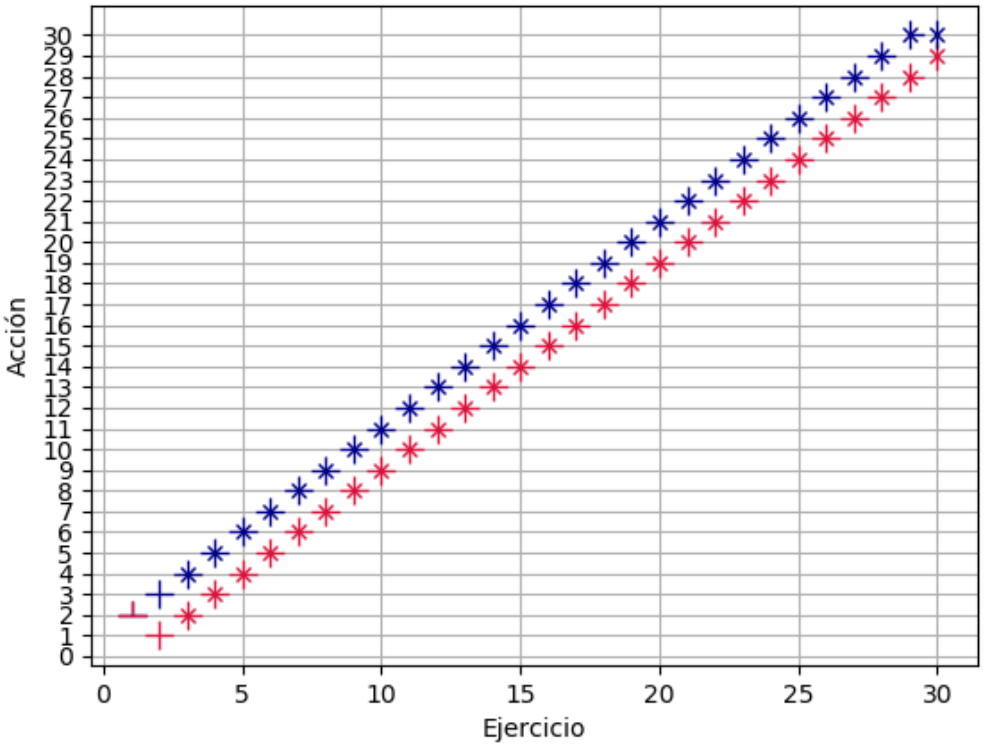

Figura 4.8: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Con lo cual vemos que la prueba 2 se comporta de manera similar a la prueba 1 (caso dependiente solo de un estado). Luego, se continuó utilizando la familia de probabilidades que considera los dos estados como parámetros, obteniendo el resultado de la prueba 3, en la cual se encuentra la misma táctica óptima resultante para los casos previos.

Prueba 3:

- Nivel = 30
- Tiempo: $\sim$ 0.387s
- Iteraciones: 2-2-2-2-2-2 (6 generales)
- Valor V: 5.657

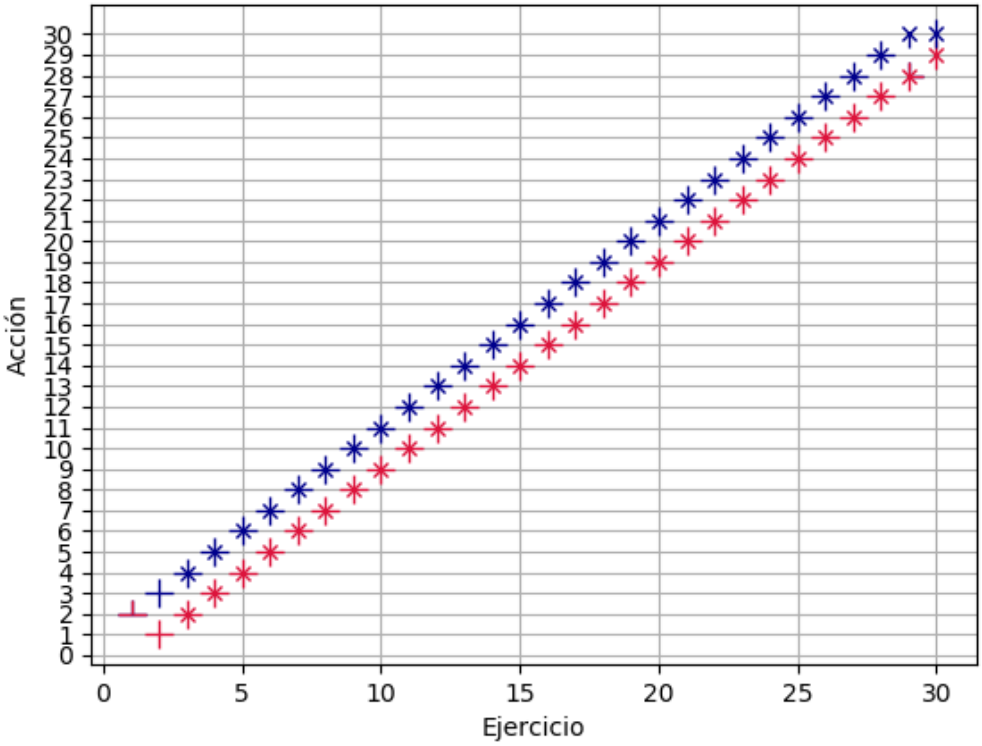

Figura 4.9: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

4.4.2. Modelo Monte Carlo

Este método es estocástico, por lo que a pesar de mantener los mismos parámetros entre una prueba y otra, no solo el tiempo puede variar y de manera considerable, sino que también las iteraciones generales, el valor de V, y también la táctica π resultante.

Como este método calcula directamente el valor V aproximado de v_π a partir de datos obtenidos de una simulación, pero no existen sub-iteraciones como pasaba con el otro método, entonces hay solo iteraciones generales asociadas a la cantidad de veces que se iteró para recalcular la táctica π en la prueba.

Además como el parámetro Ronda indica que tan buen estimador es V de v_π, ya que $\lim_{\text{Ronda}\to\infty} V_{\text{Ronda}} = v_\pi$, es decir, a más alto el valor de Ronda, se obtiene un mejor estimador asociado. Por lo que después se probará aumentando este parámetro para tener mejores estimadores, por lo que se utilizaron valores de 200, 1000 y 1500.

Y por último, notemos que mientras más alto sea el parámetro N, el problema asociado a resolver será de mayor complejidad, y más costoso computacionalmente. Entonces para realizar algunas pruebas y llegar a que el algoritmo convergiese a algo, se hizo necesario bajar el valor del parámetro N de 30 a 10.

A continuación se utilizará $N = 30$ en las pruebas de 1 al 6.

Táctica dependiente del estado previo

Prueba 1:

- $\varepsilon_M = 0.001$
- Ronda = 200
- Nivel = 30
- Tiempo: $\sim$ 0.3s
- Iteraciones: $\sim$ 2
- Valor V: $\sim$ 5

Prueba 2:

- $\varepsilon_M = 0.001$
- Ronda = 1000
- Nivel = 30
- Tiempo: $\sim$ 33s
- Iteraciones: $\sim$ 2
- Valor V: 4.9 $\sim$ 5.2

Prueba 3:

- $\varepsilon_M = 0.001$
- Ronda = 1000
- Nivel = 30
- Tiempo: 98 $\sim$ 112s
- Iteraciones: 2 $\sim$ 5
- Valor V: 4.6 $\sim$ 5.7

Observación Haciendo otras pruebas, se observa que en general el valor V va subiendo a medida que ocurren más iteraciones o el parámetro *Ronda* es mayor.

Las tres pruebas anteriores produjeron el mismo resultado en la táctica resultante, que se verá representado en el siguiente gráfico:

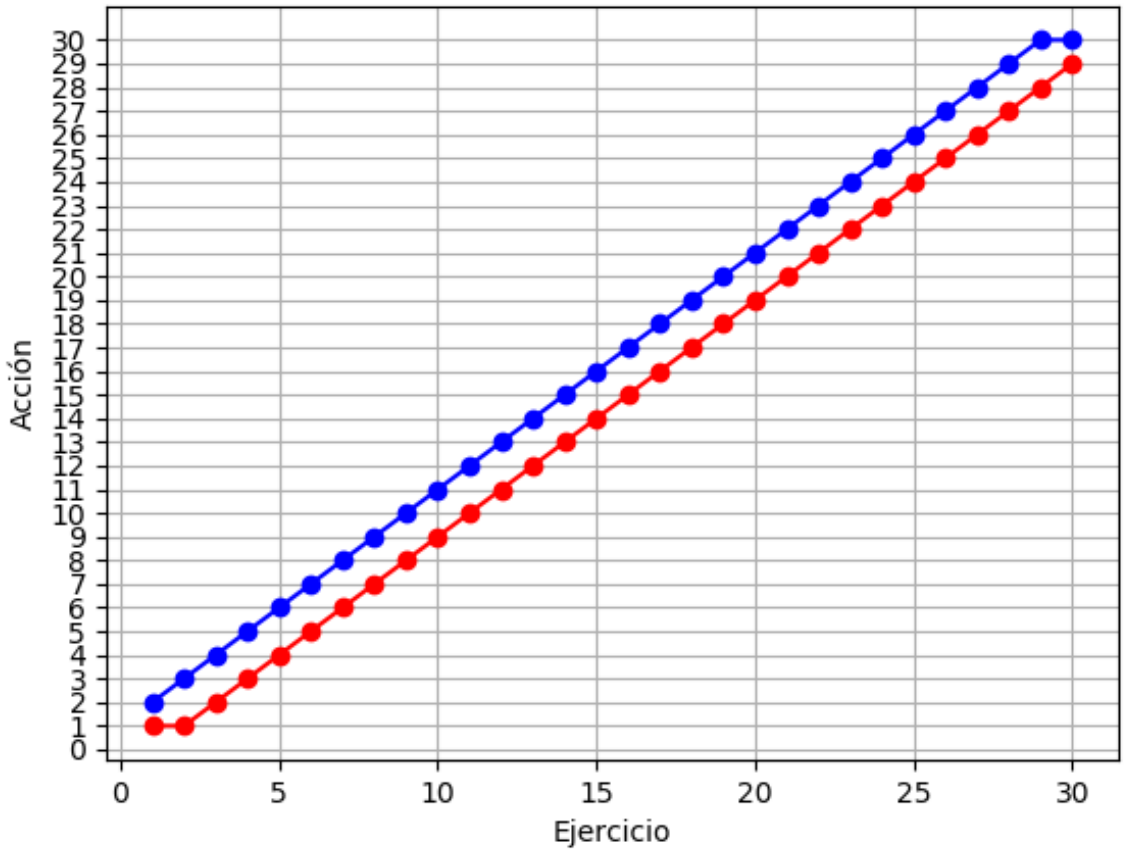

Figura 4.10: Gráfico de la táctica resultante en las pruebas 1, 2 y 3, mostrando la acción elegida por nivel de ejercicio

Táctica dependiente de los dos estados previos

Al igual que se hizo previamente, se evalúa el algoritmo utilizando la familia de probabilidades usadas para el caso de un estado.

Prueba 4:

- $\varepsilon_M = 0.001$
- Ronda = 200
- Nivel = 30
- Tiempo: $\sim$ 6.7s
- Iteraciones: $3 \sim 5$
- Valor V: $\sim$ 6.5

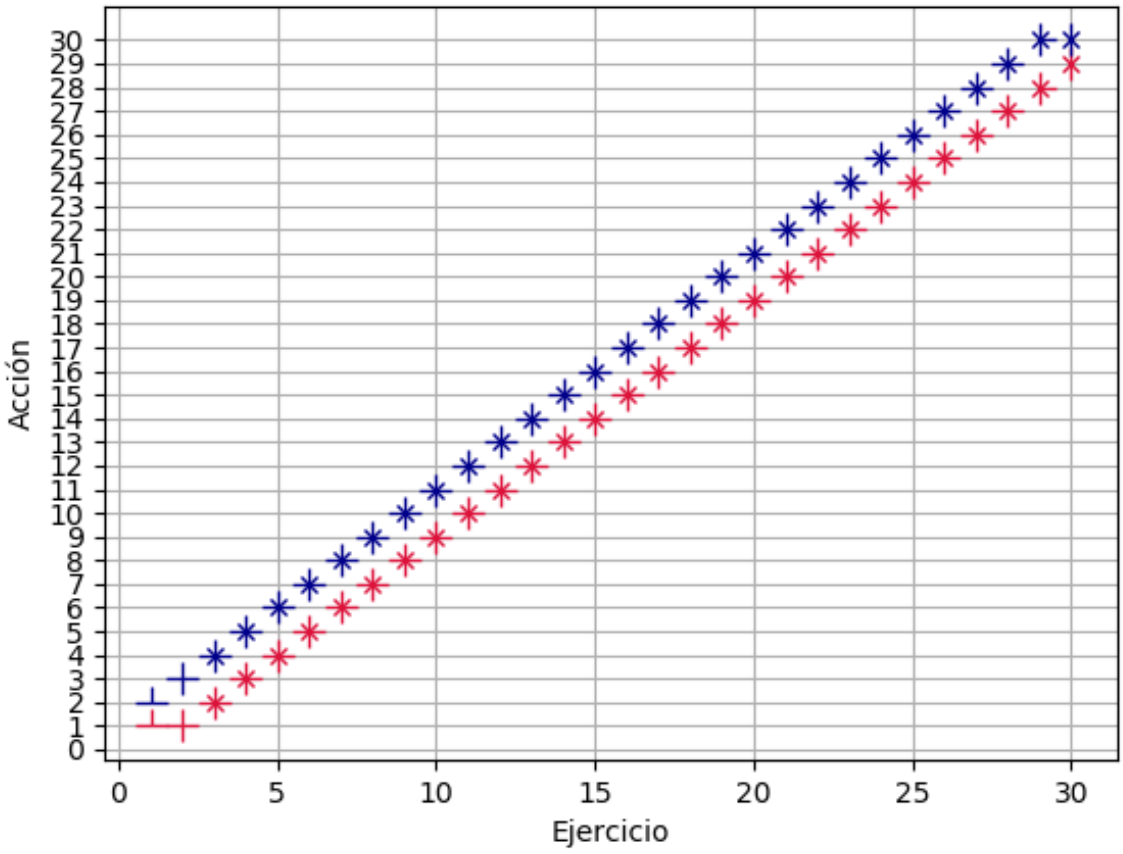

Figura 4.11: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Con lo cual vemos que la prueba 4 se comporta de manera similar a la prueba 1 (caso dependiente solo de un estado). Luego, se continuó utilizando la familia de probabilidades que considera los dos estados como parámetros, se obtienen los resultados para las pruebas 5 y 6, encontrando la misma táctica óptima resultante para ambos casos:

Prueba 5:

- $\varepsilon_M = 0.001$
- Ronda = 200
- Nivel = 30
- Tiempo: $6 \sim 6.5$s
- Iteraciones: $3 \sim 5$
- Valor V: ~ 7

Prueba 6:

- $\varepsilon_M = 0.001$
- Ronda = 1500
- Nivel = 30
- Tiempo: ~ 124s
- Iteraciones: $3 \sim 5$
- Valor V: $7 \sim 8$

Observación Si a estas pruebas se le agrega un tope mínimo de 5 iteraciones para los respectivos cálculos, la táctica resultante es la misma que la obtenida desde la iteración 2 o 3 hacia adelante, y el V respectivo no varia mayormente.

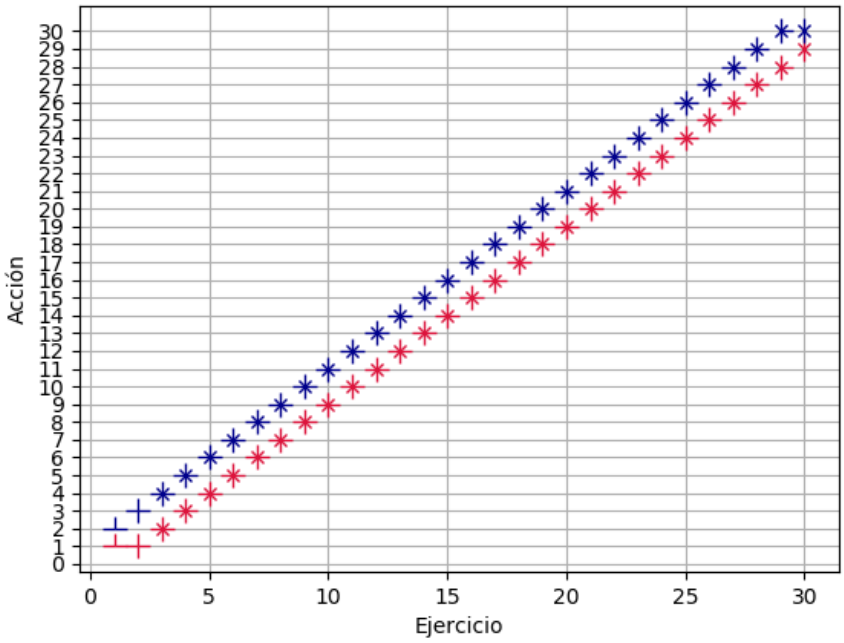

Figura 4.12: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

4.5. Resultados con variación en la recompensa

En general en los modelos del Reinforcement Learning son parámetros muy importantes las funciones de probabilidad que se usan para el entorno con que se interactúa, y las funciones de recompensa. Encontrar buenos parámetros para estas funciones de tal forma que modelen de buena manera el problema, resulta un desafío complejo, porque una pequeña variabilidad, puede producir cambios drásticos. Respecto a estos mismos cambios, es que se vio que por ejemplo en el caso de las tácticas dependientes de un estado, al cambiar ligeramente la función de probabilidad, asignando más peso a que $s_{t+1} = a_t$, esta perdía estabilidad, no convergiendo aunque se intentaran modificar otros parámetros para regular su complejidad.

Es por lo anterior expuesto, es que resulta interesante estudiar que ocurre al modificar ligeramente la función de recompensa. Para lo cual se considera una pequeña variante en la recompensa cuando la respuesta a un ejercicio es negativa (otorgando una recompensa nula si la acción es igual al estado en que se está, en vez de otorgar una recompensa de -1 como era antes), quedando como:

r(s,a,o)	$s < a$	$a \leq s$
$o = 0$	-1	0

r(s,a,o)	$s < a$	$a \leq s$
$o = 1$	2	1

Tabla 4.5: Función de recompensa modificada

Se obtienen los resultados que se mostrarán a continuación, para los cuales se usó la misma notación que la expuesta anteriormente, ya sea en la presentación de los datos, como en la motivación para modificar ciertos parámetros en ciertas pruebas.

4.5.1. Programación Dinámica

Táctica dependiente del estado previo

- Nivel = 30
- Tiempo: $\sim$ 0.37s
- Iteraciones: 20-11-1 (3 generales)
- Valor V: 5.9

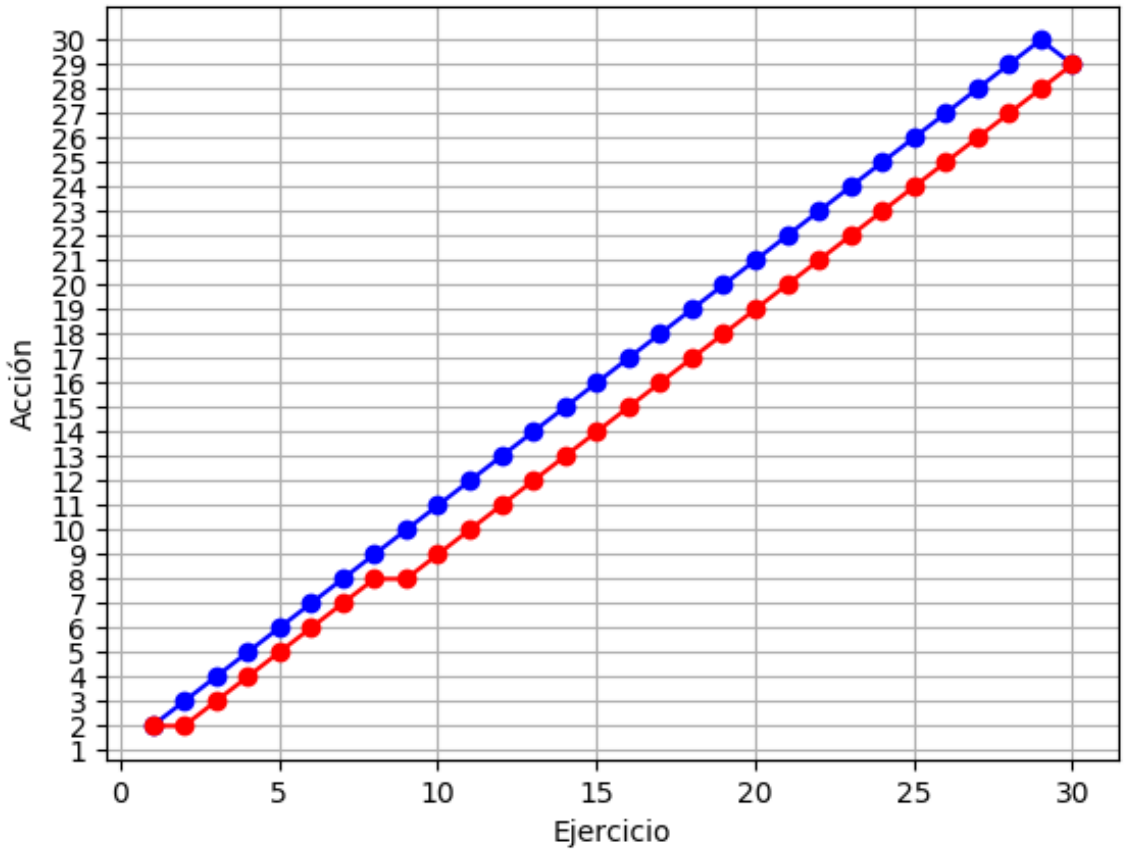

Figura 4.13: Gráfico de táctica, mostrando la acción elegida por nivel de ejercicio.

Táctica dependiente de los dos estados previos

Dada una táctica, anteriormente se calculó el valor de V de manera exacta. Pero con los cambios hechos en esta sección, esto produjo problemas con su convergencia por lo que se tuvo que incorporar un error (o considerar $\varepsilon_E > 0$) para así permitir que el algoritmo se detuviese.

Comenzando con el procedimiento hecho anteriormente, se evalúa el algoritmo utilizando la familia de probabilidades usadas para el caso de un estado.

Prueba 1:

- $\varepsilon_E = 0{,}005$
- Nivel = 30
- Tiempo: $\sim$ 0.2s
- Iteraciones: 2-2-2 (3 generales)
- Valor V: 4.313

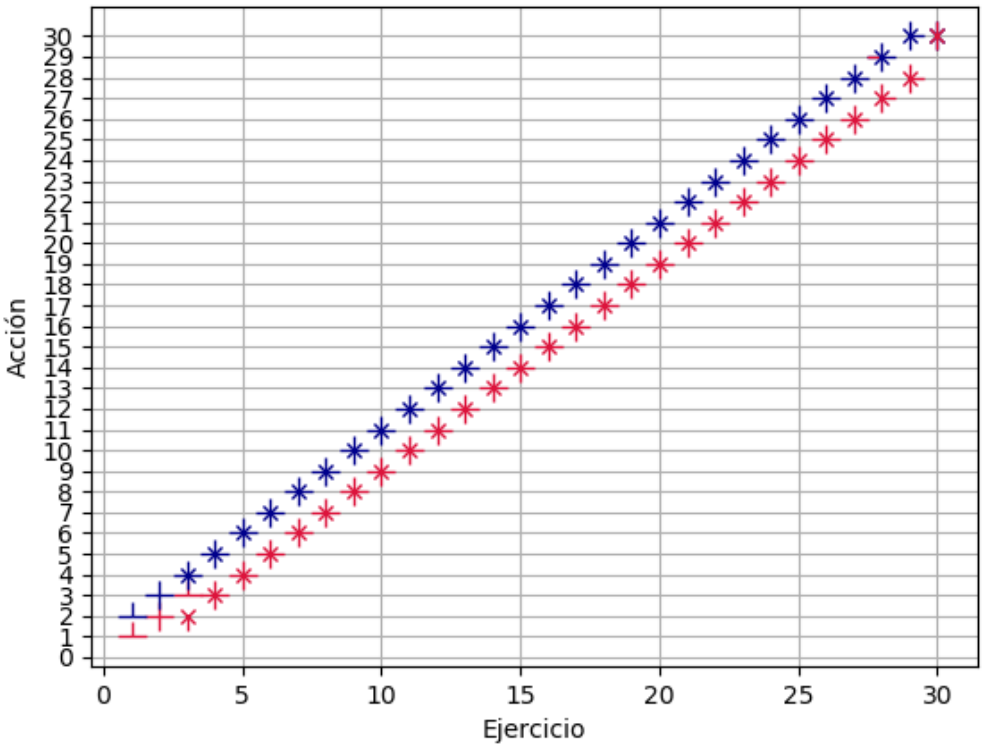

Figura 4.14: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Prueba 2:

- $\varepsilon_E = 0{,}001$
- Nivel = 10
- Tiempo: $\sim$ 0.1s
- Iteraciones: 2-2-2-2 (4 generales)
- Valor V: 4.76

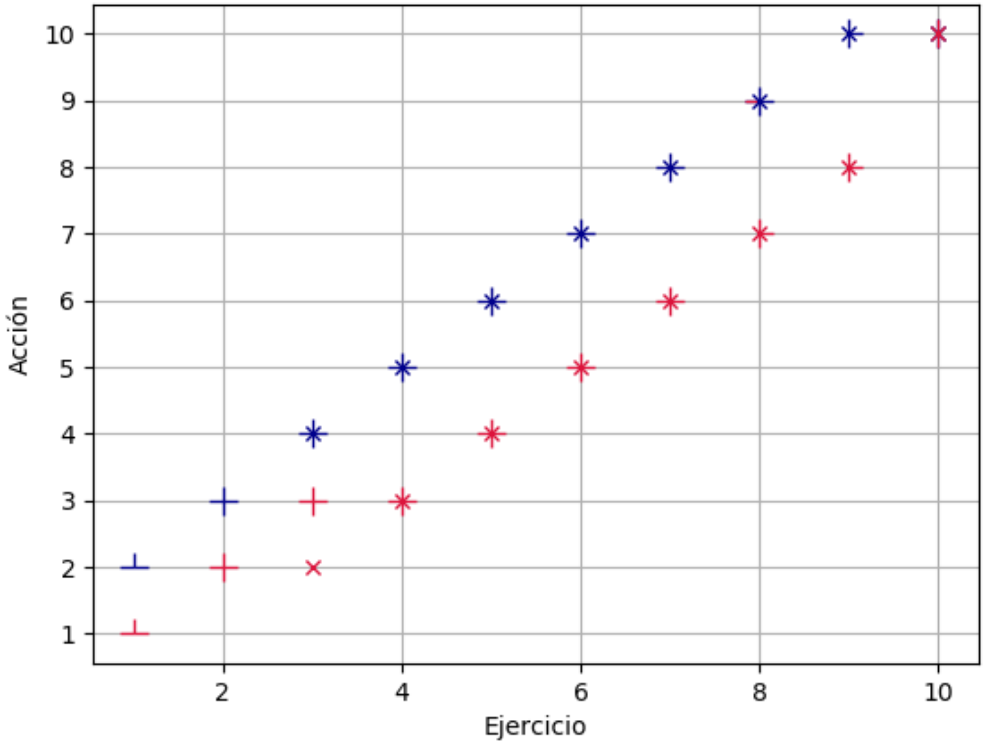

Figura 4.15: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Prueba 3:

- $\varepsilon_E = 0{,}0001$
- Nivel = 10
- Tiempo: ~ 0.21s
- Iteraciones: 2-2-2-2-2-2-2-2 (8 generales)
- Valor V: 5.53

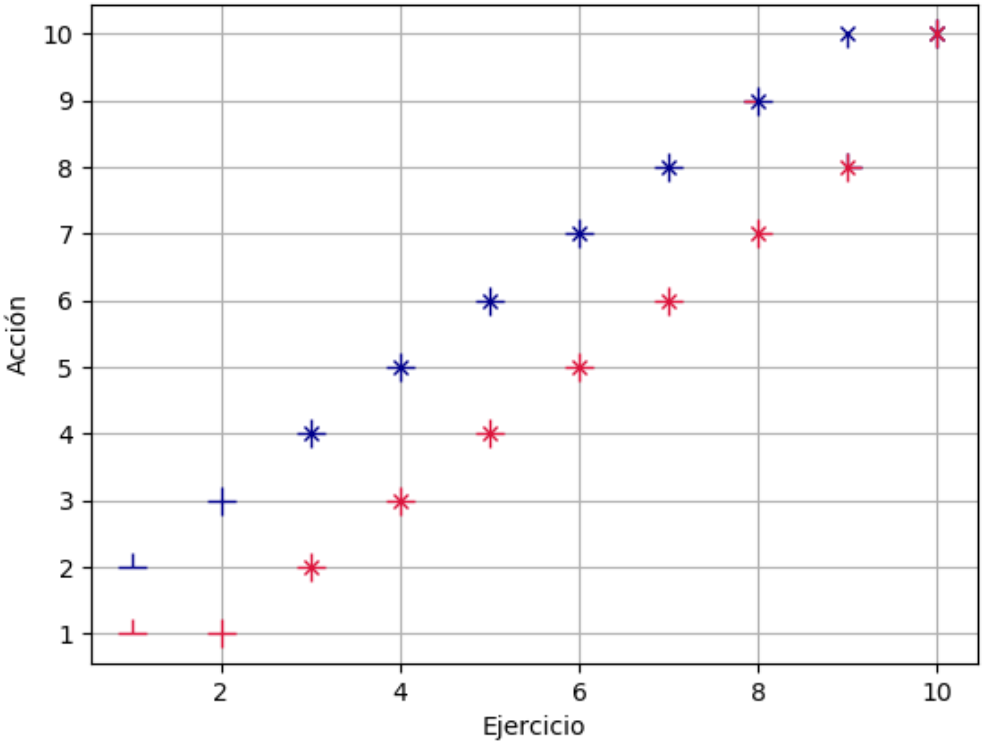

Figura 4.16: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Con esto se observan resultados similares al caso de un estado, por lo que proseguimos usando la familia de probabilidades que considera los 2 estados.

Prueba 4:

- $\varepsilon_E = 0{,}005$
- Nivel = 30
- Tiempo: $\sim$ 0.433s
- Iteraciones: 2-2-2-2-2-2 (6 generales)
- Valor V: 5.68

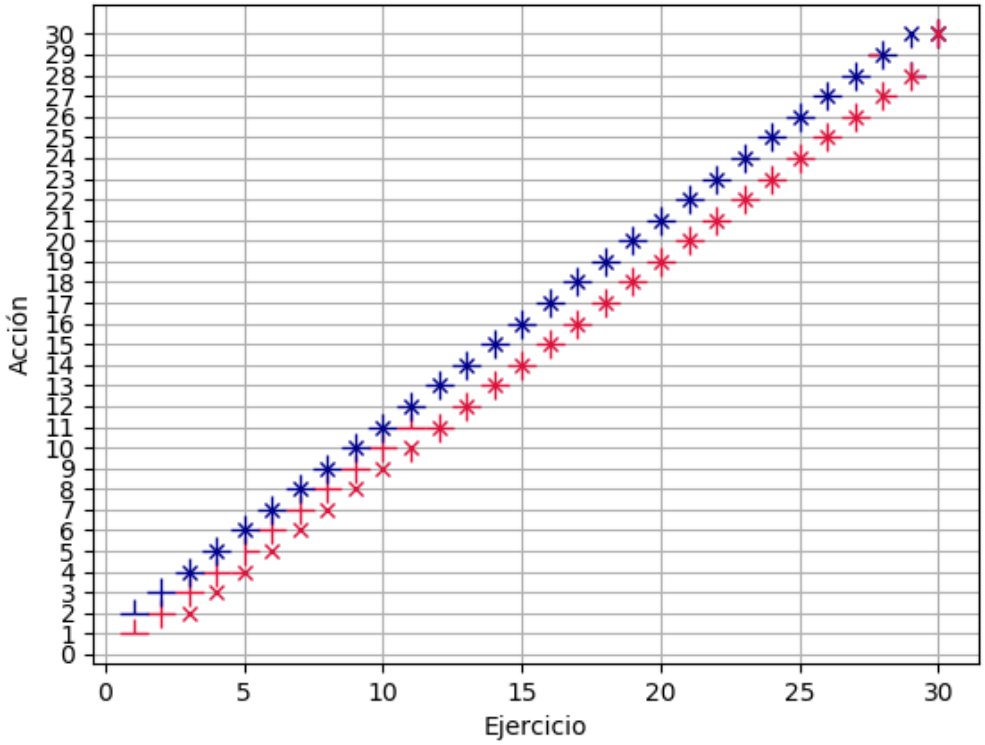

Figura 4.17: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Prueba 5:

- $\varepsilon_E = 0{,}001$
- Nivel = 10
- Tiempo: $\sim$ 0.2s
- Iteraciones: 2-2-2-2-2-2-2 (7 generales)
- Valor V: 5.424

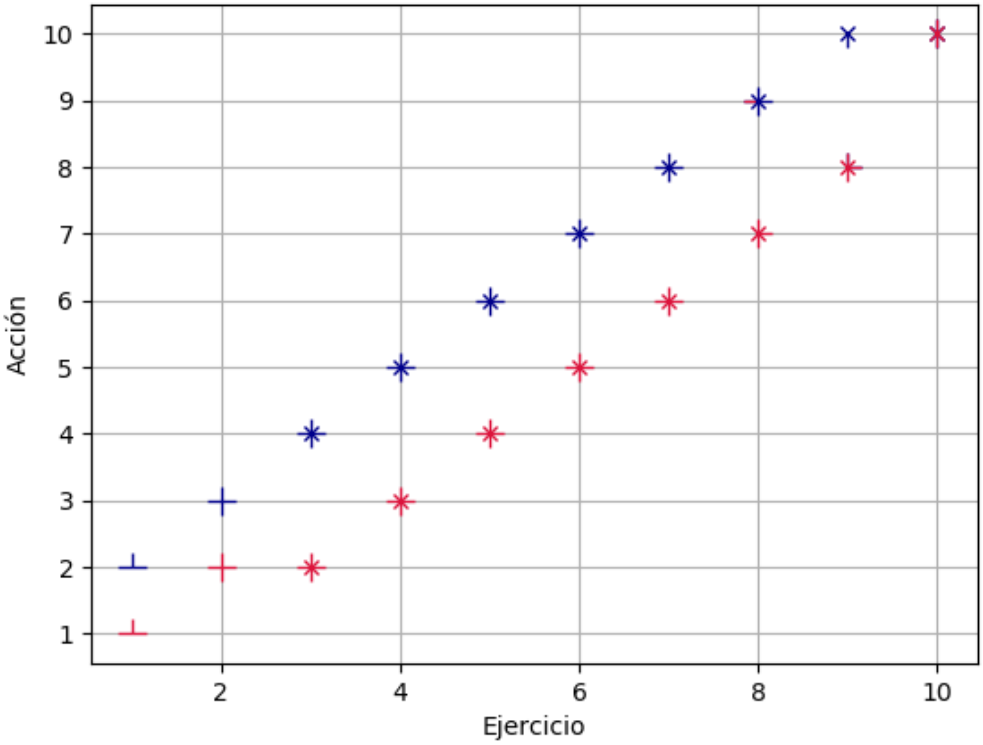

Figura 4.18: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Prueba 6:

- $\varepsilon_E = 0{,}0001$
- Nivel = 10
- Tiempo: $\sim$ 0.243s
- Iteraciones: 2-2-2-2-2-2-2-2-2-2 (10 generales)
- Valor V: 5.6

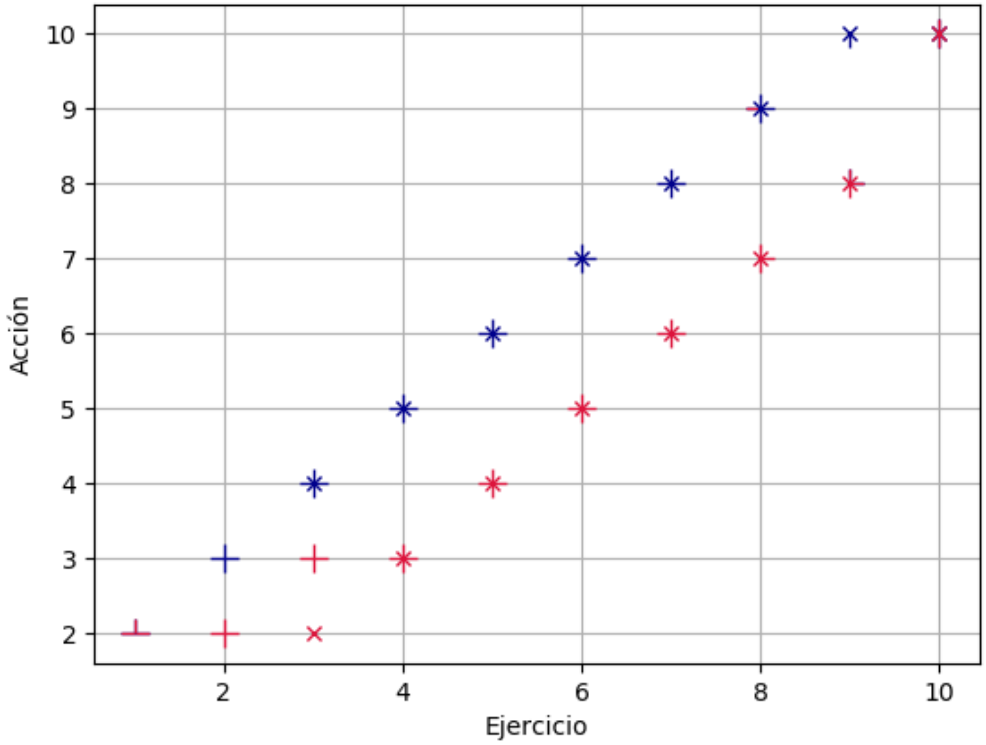

Figura 4.19: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

4.5.2. Modelo Monte Carlo

Táctica dependiente del estado previo

Para estos casos, con los parámetros originales, el algoritmo no terminaba en un tiempo prudente comparado con las otras pruebas (después de 1 hora seguía iterando). Por lo que fue necesario ir ajustando los parámetros de error ε_M, valor de la ronda y niveles (valores omitidos, se asumen como usados los valores base presentados al comienzo de esta sección). Es así como se fue subiendo el ε_M hasta el valor de 0.1 , para lograr que el algoritmo terminara, obteniendo un resultado y así tener una base de la que partir buscando mejorar estos. Monte Carlo es un método exploratorio, por lo que da muchas vueltas en las tácticas que puedan ser útiles, y al trabajar con Nivel = 30 se tiene que el conjunto de tácticas posibles es demasiado grande ($\sim 3^{30}$), por lo que se bajó a Nivel = 10, y así bajar considerablemente este conjunto factible de tácticas, además de subir el valor de Ronda, para que en cada iteración se logre una mejor estimación de v_π.

Usando $\varepsilon_M = 0.1$ en las pruebas del 1 al 6.

Prueba 1 (se puso tope mínimo de 5 iteraciones):

- $\varepsilon_M = 0.1$
- Ronda = 200
- Nivel = 10
- Tiempo: ~ 0.32s
- Iteraciones: ~ 5
- Valor V: $4.5 \sim 5$

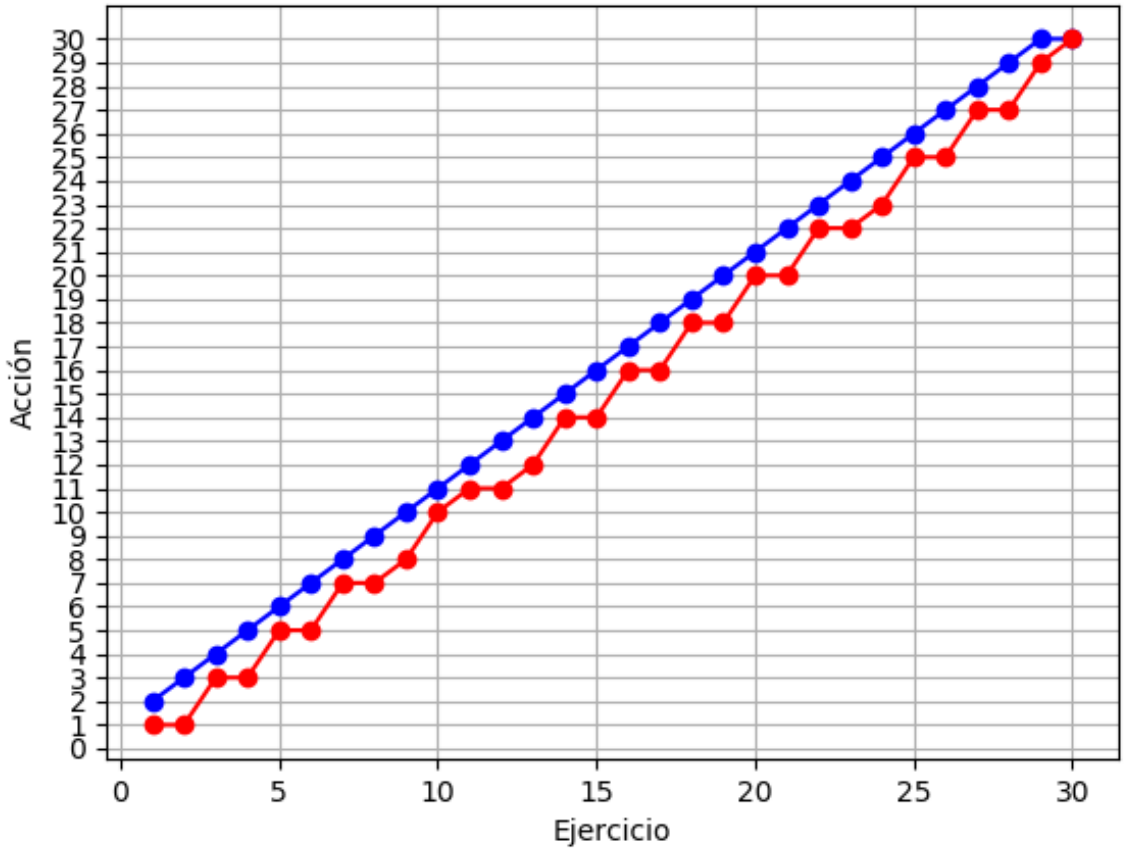

Figura 4.20: Gráfico de táctica de la prueba 1, mostrando la acción elegida por nivel de ejercicio.

Prueba 2

- $\varepsilon_M = 0.1$
- Ronda = 1000
- Nivel = 10
- Tiempo: $\sim$ 35s
- Iteraciones: $\sim$ 5
- Valor V: 4 $\sim$ 5

Prueba 3:

- $\varepsilon_M = 0.1$
- Ronda = 1000
- Nivel = 10
- Tiempo: $\sim$ 31s
- Iteraciones: $\sim$ 5
- Valor V: $\sim$ 5.2

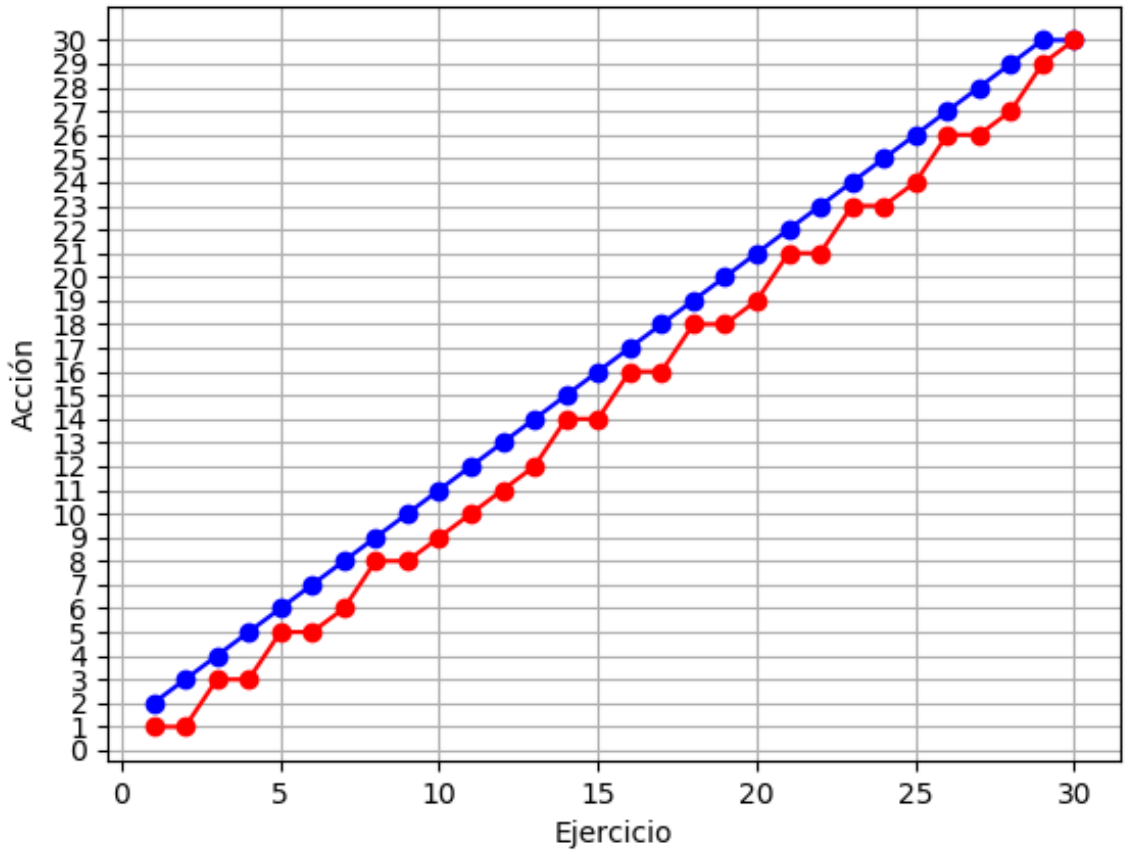

Figura 4.21: Gráfico de táctica de las pruebas 2 y 3, mostrando la acción elegida por nivel de ejercicio.

En las pruebas siguientes, comienza a ser relevante el corte por táctica repetida (detección de loop en las tácticas encontradas no contiguas en iteraciones), es decir, se da la posibilidad de que el algoritmo termine sin considerar el error fijado como parámetro.

Usando $\varepsilon_M = 0.01$, Ronda = 1000 y Nivel = 10 en las pruebas 4 y 5.

Prueba 4 (termina sin loop):

- $\varepsilon_M = 0.01$
- Ronda = 1000
- Nivel = 10
- Tiempo: $\sim$ 43s
- Iteraciones: $\sim$ 6
- Valor V: $\sim$ 5.3

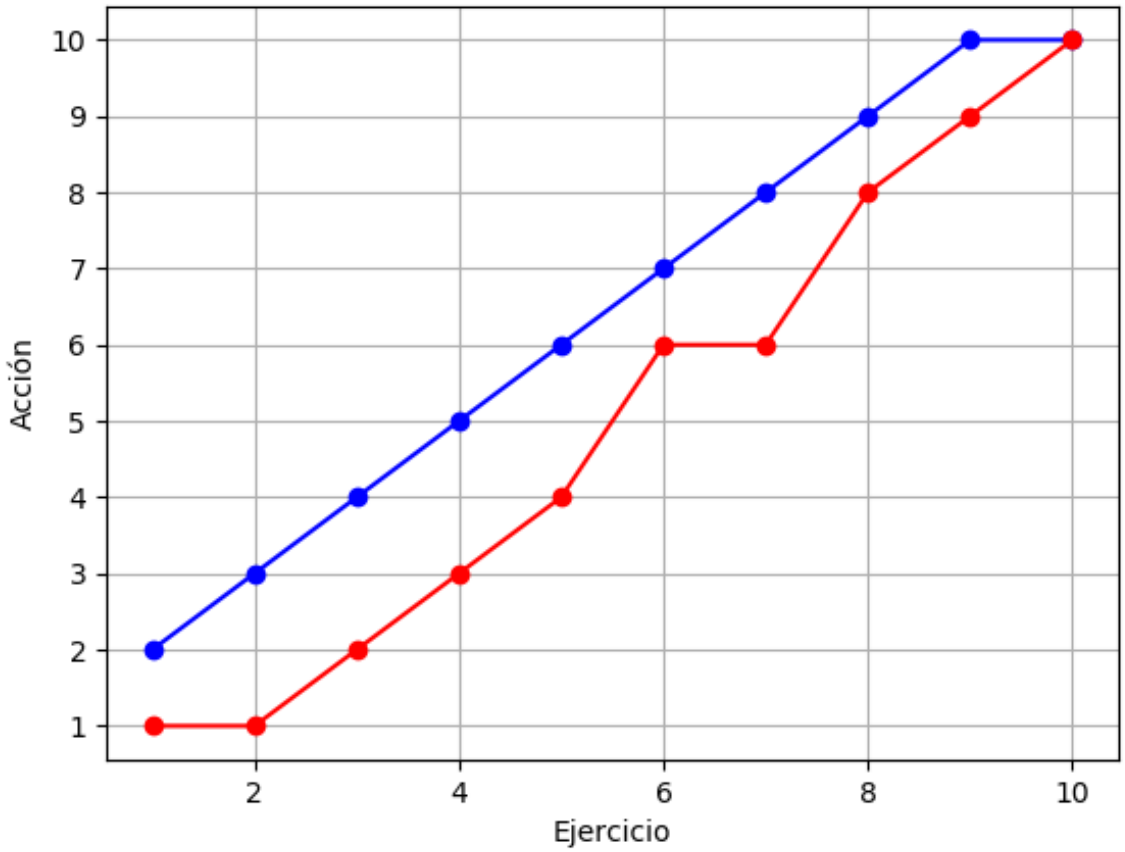

Figura 4.22: Gráfico de táctica de la prueba 4, mostrando la acción elegida por nivel de ejercicio.

Prueba 5 (termina con loop):

- $\varepsilon_M = 0.01$
- Ronda = 1000
- Nivel = 10
- Tiempo: $\sim$ 38s
- Iteraciones: $\sim$ 5
- Valor V: $\sim$ 7.57

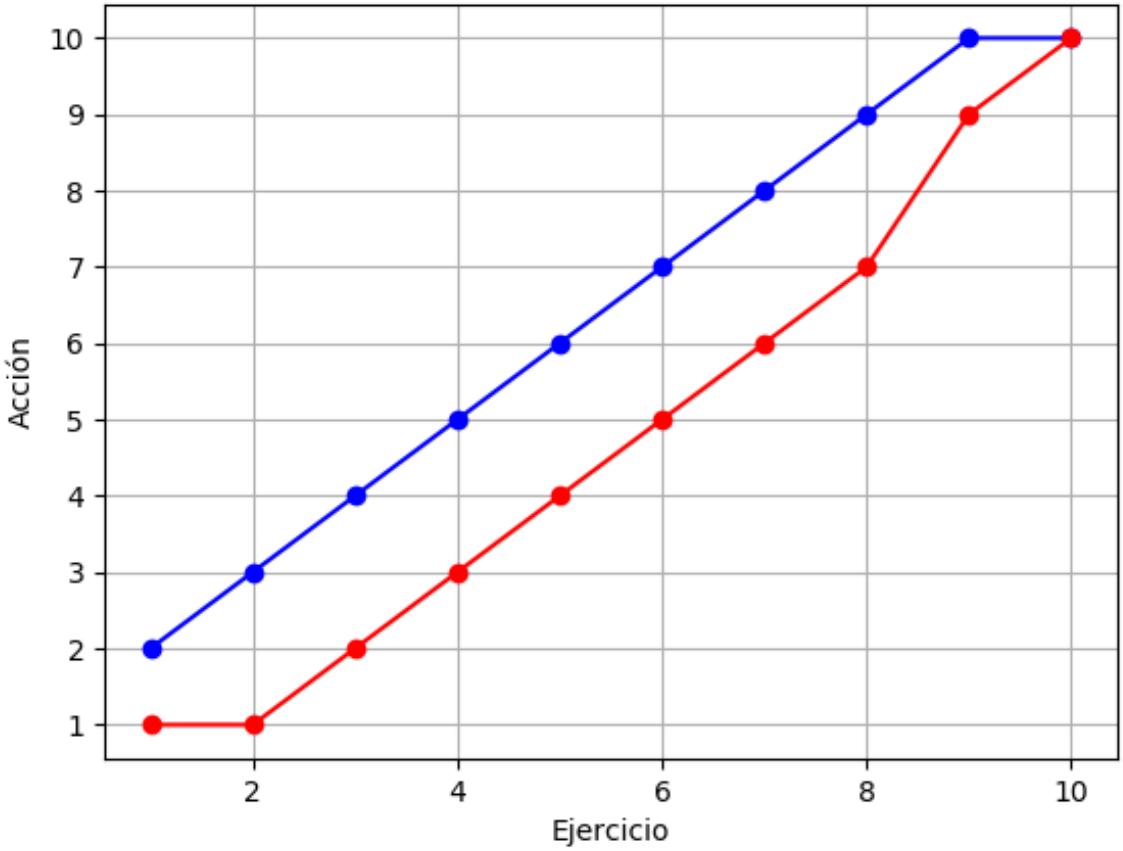

Figura 4.23: Gráfico de táctica de la prueba 5, mostrando la acción elegida por nivel de ejercicio.

Prueba 6 (termina sin loop):

- $\varepsilon_M = 0.001$
- Ronda = 1500
- Nivel = 10
- Tiempo: $\sim$ 165s
- Iteraciones: $\sim$ 18
- Valor V: $\sim$ 3.9

Prueba 7 (termina con loop):

- $\varepsilon_M = 0.001$
- Ronda = 1000
- Nivel = 10
- Tiempo: 63 $\sim$ 96
- Iteraciones: 10 $\sim$ 14
- Valor V: 3.4 $\sim$ 4.2

Prueba 8 (solo para con loop):

- $\varepsilon_M = 0.001$
- Ronda = 1500
- Nivel = 10
- Tiempo: $\sim$ 181s
- Iteraciones: $\sim$ 9
- Valor V: $\sim$ 4.9

Táctica dependiente de los dos estados previos

Al igual que anteriormente, primero se evaluó el algoritmo restringiéndolo en su dependencia en las probabilidades a un estado (se utilizó la familia de probabilidades usadas para el caso de un estado), obteniendo:

Prueba 1 (semejante a la prueba 1 de la parte anterior):

- $\varepsilon_M = 0.1$
- Ronda = 200
- Nivel = 30
- Tiempo: $7 \sim 7.5$s
- Iteraciones: ~ 5
- Valor V: $5 \sim 7$

Prueba 2 (semejante a las pruebas 2 y 3 de la parte anterior):

- $\varepsilon_M = 0.1$
- Ronda = 1000
- Nivel = 10
- Tiempo: $33 \sim 103$s
- Iteraciones: $5 \sim 15$
- Valor V: $5.5 \sim 8$

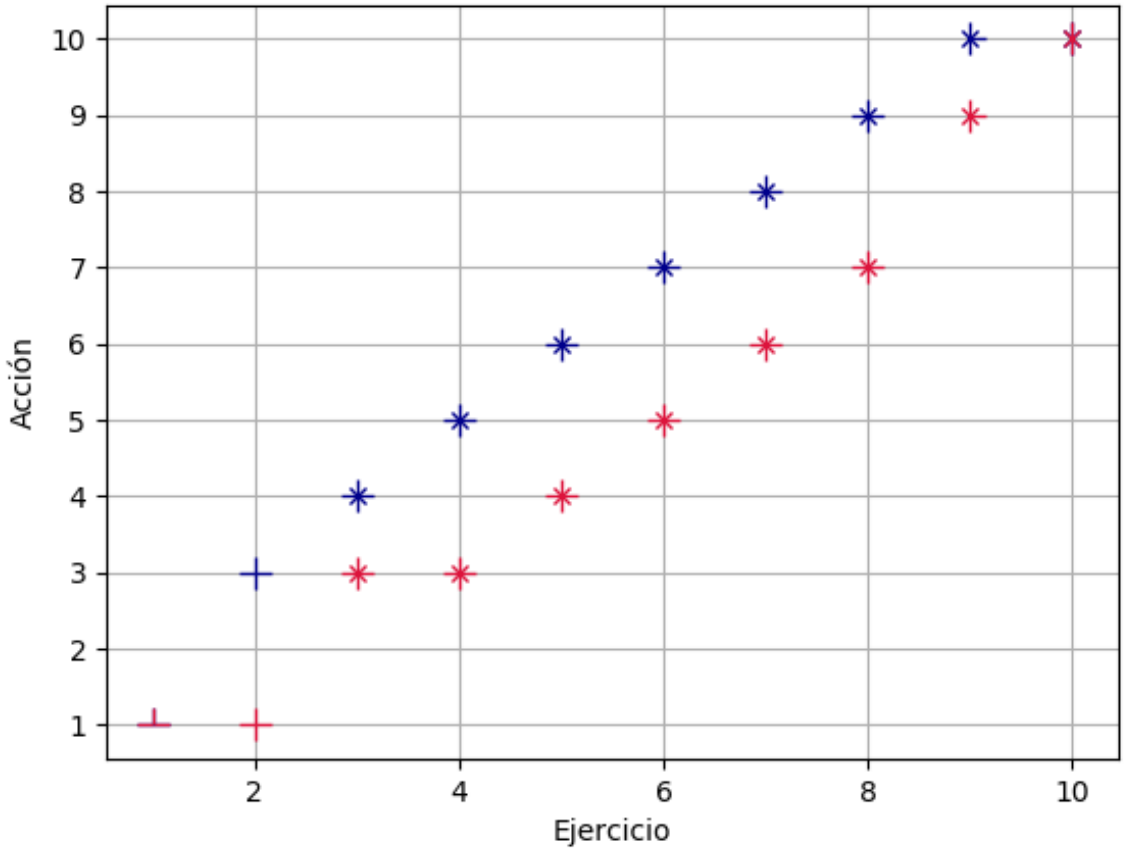

Figura 4.24: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

En las pruebas 1 y 2, los más rápidos terminaron sin loop, pero muchos terminan con loop.

Al igual que se hizo anteriormente, se consideró la familia de probabilidades que contempla los dos estados como parámetros. Además, debido a que el algoritmo no logra encontrar el óptimo en un tiempo razonable (menor a 15 minutos), se relaja el problema como para el caso de 1 estado, aumentando el valor de ε_M en las pruebas 3 y 4.

Prueba 3 (se puso tope mínimo de 5 iteraciones):

- $\varepsilon_M = 0.1$
- Ronda = 200
- Nivel = 30
- Tiempo: $\sim$ 7s
- Iteraciones: 2 $\sim$ 5
- Valor V: $\sim$ 7

Se obtiene una táctica que se muestra en el siguiente gráfico.

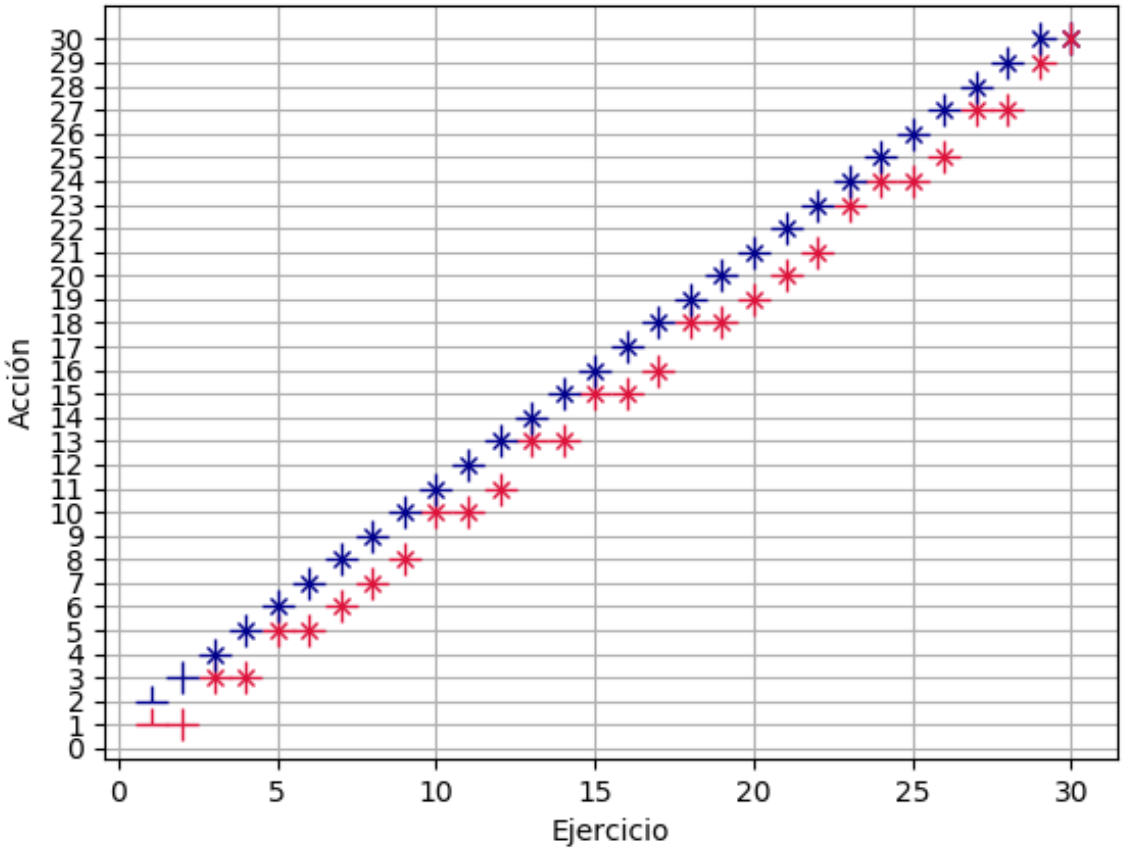

Figura 4.25: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Luego se buscó quitar parcialmente la relajación, para esto se usó $\varepsilon_M = 0.01$, además para potenciar la convergencia se utiliza Ronda = 1000 y para disminuir la complejidad del problema, utilizaremos Nivel = 10.

Prueba 4 (se puso tope mínimo de 5 iteraciones):

- $\varepsilon_M = 0.01$
- Ronda = 1000
- Nivel = 10
- Tiempo: $54 \sim 90$s
- Iteraciones: $8 \sim 14$
- Valor V: ~ 8.8

Se consigue el siguiente resultado como ejemplo de una de las muestras tomadas.

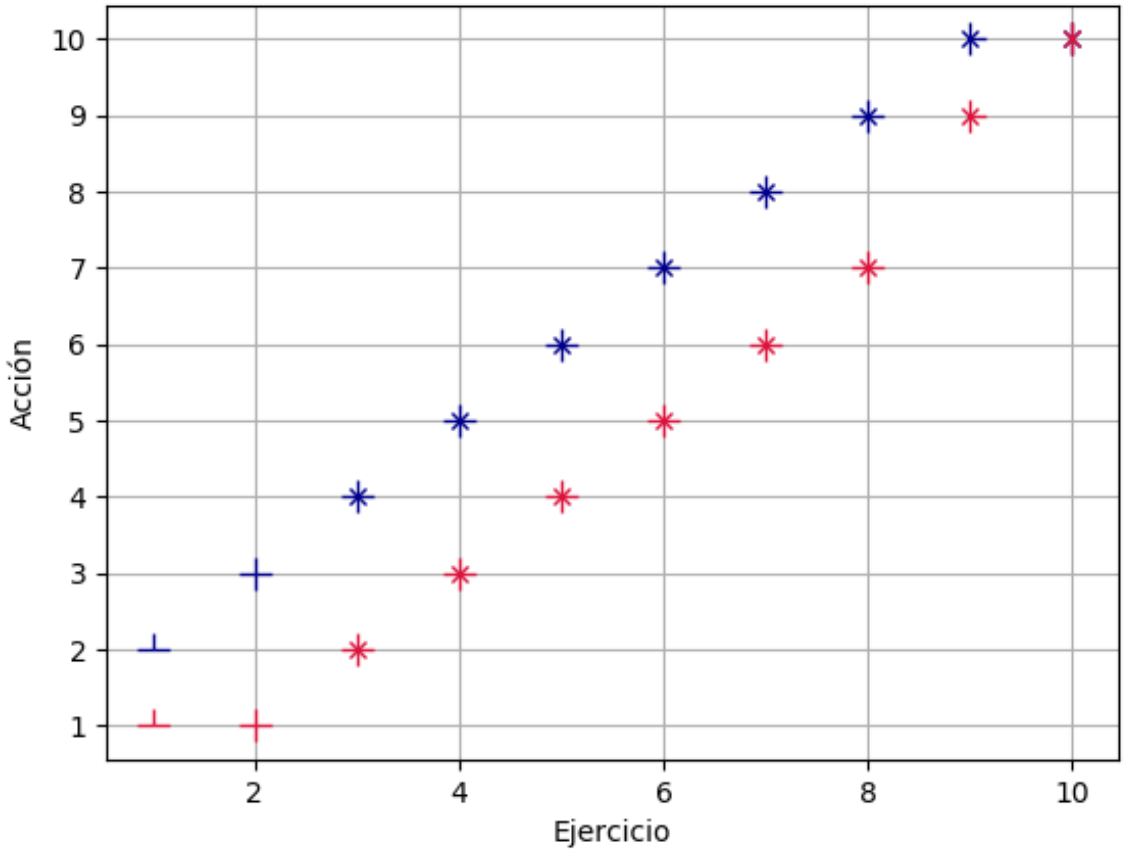

Figura 4.26: Gráfico de táctica, mostrando la acción elegida por niveles de ejercicio.

Capítulo 5

Análisis y discusión

En este capítulo se analizarán los resultados obtenidos en el capítulo 4, y se discutirán aspectos que parezcan relevantes de mencionar. Se dividirá este capítulo en dos secciones, analizando así por separado los dos objetivos principales de este trabajo, considerando que por un lado está el simulador de estudiante usado para los cálculos internos y probar las tácticas resultantes, y por el otro lado están los métodos expuestos utilizados para obtener las tácticas.

5.1. Análisis y discusión de simulador de estudiante

A partir de resultados de las simulaciones mostradas más arriba en los gráficos de la sección 4.3, se puede observar que:

- En el gráfico de la figura 4.4 que representa un ejemplo de como varía la probabilidad de acierto a lo largo del avance en el flujo de ejercicios, se puede apreciar que existen variaciones de aproximadamente un 20 % de probabilidad absoluta luego de errar ejercicios, en donde el aumento abrupto es un problema, y el descendente estabiliza el modelo y vuelve a la tendencia. En cambio, entre los momentos 29 al 32 no hay cambios abruptos de probabilidad de acierto. Lo que se interpreta del problema de las variaciones de la probabilidad es que se asume que un estudiante que erró un ejercicio, es prácticamente improbable que vuelva a errar otro del mismo nivel o inferior. Esto fue algo que se quiso evitar por su poca proximidad al comportamiento real de un estudiante. Como potencial trabajo futuro, es recomendable realizar ajustes a la función de probabilidad de acierto.

- Una de las ideas centrales es que existen niveles que una vez superados por el alumno se presume que no van a bajar de nivel. Por lo tanto, se incorporó una restricción en el modelo del simulador del estudiante en el nivel 29, que enrostra esta presunción, lo cual es útil y necesario para evitar que se den los casos en que a pesar de que el estudiante simulado tuviese un buen desempeño, fuera bajando de nivel solo por la probabilidad de las acciones a considerar. A partir de lo anterior, el resultado en el gráfico de la figura 4.6 se aprecia una cuña entre los niveles 28, 29 y 30. En donde por ejemplo, en la curva

correspondiente al *Momento 4* la probabilidad del nivel 29 es más baja que la del nivel 28, esto se debe a que cuando hay respuestas incorrectas, el siguiente nivel de ejercicio siempre será el 29 (pero sin la restricción puesta también podría ser que toque el nivel 28), asociado a una probabilidad de éxito más baja al contestar el siguiente ejercicio, bajando la media de probabilidad del nivel 29, por lo tanto, se justifica la formación de la cuña.
En este estudio, se incluyó la restricción para simular la realidad esperada de una secuencia de aprendizaje progresivo, pero resulta interesante estudiar estos mismos resultados alterando las probabilidades en el modelo.

- En el gráfico de la figura 4.6, para el análisis es conveniente considerar los resultados a partir del *Momento 30*, porque en promedio los estudiantes simulados ya habrán pasado al menos una vez por los 30 niveles de ejercicio. A partir de ese punto, la probabilidad de acierto disminuye a medida que aumenta el momento y a medida que aumenta el nivel de ejercicio. Este resultado es esperable, porque los estudiantes con un menor desempeño llegan a los niveles de ejercicio más altos en los últimos momentos estudiados, disminuyendo la probabilidad de acierto general. Es importante destacar que a pesar de que las simulaciones se enfocan en un comportamiento individual, se aprecia como se logra un comportamiento esperable a nivel global.

5.2. Análisis y discusión de resultados de los métodos expuestos

A modo general los tiempos promedios de los métodos aplicados a las tácticas dependientes de 2 estados son mayores que los tiempos promedios de las tácticas dependientes de 1 estado. Estos resultados eran esperables, debido al incremento del tamaño del conjunto factible de las tácticas, lo que hace aumentar la complejidad de resolución del problema.

Recordar que las funciones de valor (V) permiten encontrar la táctica que tenga el valor más alto. Por lo tanto, estas resultan ser un instrumento intermedio para obtener el producto final del problema que es obtener una táctica óptima. Si bien, teniendo en cuenta que muchos casos los valores de V asociados a las tácticas parciales de 2 estados obtenidas son menores a las de 1 estado, esto no quiere decir que una táctica sea mejor que la otra o que la diferencia entre los valores de V indique que una de ellas no sea óptima, básicamente porque son 2 problemas distintos no comparables en ese sentido, por lo tanto los mecanismos de obtención de las tácticas aplicado al caso de 2 estados puede que no sean eficientes, pero sí funcionan. Ahondando en esto, se puede destacar que se obtienen tácticas similares entre ambos casos, teniendo principalmente variantes en los extremos, debido a las condiciones de borde incorporadas en el algoritmo para que tenga una buena partida (primeras respuestas no influyen significativamente aunque sean negativas, incentivando el avance y no quedarse pegado cercano a los primeros niveles) y buena detención (una vez superado el penúltimo nivel, no puede bajar de este).

Para las pruebas de tácticas dependientes de un estado realizadas en la Secciones 4.4.1 y 4.4.2, se puede observar que, utilizando los mismos parámetros, el método de MC (Monte Carlo) obtuvo el mismo resultado que el método de Programación Dinámica (PD) y encontró

la misma táctica (objetivo del problema), pero en un tiempo ligeramente mayor. Aunque su valor V asociado resulte menor y fluctuante, tiende a aumentar cuando se sube el parámetro *Ronda* asociado, acercándose lentamente al valor que se tenia en PD, sin embargo, con esto el tiempo de cómputo del algoritmo MC aumenta considerablemente haciéndolo no escalable para problemas más complejos.

En la misma Sección 4.4, al comparar las pruebas del método MC con el modelo más complejo que considera tácticas dependiente de dos estados, no se visualiza un gran cambio en las tácticas (resultan muy similares), pero sí se obtiene rápidamente un mejor valor V que para el caso de un estado para MC, e incluso que para PD, esto se da principalmente porque se está optimizando la función V sobre un conjunto más grande. Además este valor aumenta a medida que se sube el parámetro *Ronda*, dado que esto permite calcular con mayor precisión el valor de V, pero junto con un aumento considerable del tiempo de ejecución.

Un detalle interesante que podemos apreciar, es que al verificar el modelo de PD para 2 estados (PD-2) en la figura 4.8, se obtienen datos como el V y las iteraciones que son distintos a lo esperado por lo obtenido en el modelo de PD para 1 estado (PD-1) en la figura 4.7. Pero se esperaría ver el mismo resultado, ya que un método determinista realiza determinados pasos fijos o recorre el mismo camino al momento de calcular lo que se busca. Para analizar esto, primero es importante validar cada método de forma independiente, y en segundo lugar, explicar la diferencia entre resultados:

- Las bases del modelo están en los lemas 3.2 para PD-1, y 3.7 para PD-2, estas ecuaciones son propiedades que solo son satisfechas por la función v_π, ya que, se puede ver que si v_π satisface la ecuación del Lema 3.2, entonces también satisface la ecuación del Lema 3.7. Lo mismo ocurre para v^*, a partir de la Proposición 3.4 para PD-1, y la Proposición 3.9 para PD-2. Y como también estas funciones son únicas, se tiene que cada modelo es válido.

- Al calcular una aproximación, en cada modelo, presenta dos caminos distintos que convergen a un entorno de v_π, que al trabajarse computacionalmente con un error, generalmente no resulta ser el óptimo teórico (v_π). Solamente en el caso de que el tiempo de ejecución sea infinito se asegura la convergencia exacta, pero en la práctica se trabaja con tiempo acotado, en otras palabras, con un error aceptable. Por tal razón los valores de convergencia difieren entre sí. Análogamente, ocurre lo mismo para el cálculo de la aproximación de v^*.

Al verificar el modelo MC para 2 estados (MC-2) se obtienen valores distintos de V que las pruebas correspondientes al modelo MC para 1 estado (MC-1). Esto ocurre por una razón similar a la explicada anteriormente de que se obtiene la estrategia a partir de las ecuaciones base mostradas en la Proposición 3.4 para MC-1, y la Proposición 3.9 para MC-2, haciendo que se tomen dos rutas distintas de convergencia. Finalmente el resultado de ambos métodos queda sujeto al error tolerable.

Para tácticas dependientes de 1 estado correspondientes a la Sección 4.5, donde se modificó la función de recompensa de acuerdo a la Tabla 4.5, se aprecia una clara diferencia visual entre las tácticas obtenidas en MC y aquellas de PD (ver Figuras 4.20 y 4.13 respectivamente), esto se explica porque fue necesario modificar los parámetros *Nivel* y ε_M de la Sección 4.4 para

evitar tiempos de cómputo altos para MC [1]. Las modificaciones introducidas fueron un menor *Nivel* y un mayor rango de error permitido ε_M, para así lograr que convergiese en un tiempo razonable para este trabajo (entorno a los 15 minutos). Para sortear esta dificultad técnica se modificó el parámetro *Nivel* a uno menor, disminuyendo así la complejidad del problema y además, se aumentó el rango permitido de error ε_M. Como resultado, la mayoría de las tácticas no convergen claramente a una táctica especifica [2] (Figura 4.20), sin embargo las estimaciones para v_π convergen y el algoritmo se detiene cuando la diferencia entre iteraciones sucesivas en el cálculo de V resultan menores a ε_M. La convergencia en el estimador de v_π válida este resultado y permite promover una metodología exploratoria al problema, por medio de una subdivisión en problemas de menor complejidad definidos por el parámetro $N = Nivel$, esto da lugar a fases de largo N donde en cada fase se escogen acciones a partir de diferentes estrategias.

Nuevamente en relación con la Sección 4.5, en primer lugar, cuando se vuelve a probar con el valor de error original en MC-1 (4.1), disminuyendo el valor de *Nivel* y aumentando el de *Ronda* (es decir, se baja la complejidad del problema, y se aumenta el rango de simulaciones para determinar el V parcial), se obtiene que el algoritmo termina por encontrar un loop (definido en la página 56), quedando fuera del error considerado, aunque consiguiendo un resultado más parecido al de PD-1 (ver figuras 4.7 y 4.23). En segundo lugar, al verificar el modelo para el caso de MC-2, se consiguen resultados con valores de V más altos que para el caso MC-1, incluso mayores que el caso de verificación del modelo de PD-2 (ver figuras 4.24, 4.20 y 4.14 respectivamente). Además se detecta que el valor de V aumenta cuando se baja la complejidad, también se observa que al quitar holgura al error a considerar, resultan tiempos de computo considerablemente mayores, siendo esto no deseable. Ya que se tiene que las pequeñas variaciones de los parámetros en MC pueden originar problemas de convergencia, se recomienda para estudios futuros probar con otros parámetros distintos, puesto que se aprecia que desempeñan un rol importante en la eficacia del algoritmo.

[1] Se observaron tiempos de ejecución mayores a 1 hora, cantidad muy superior a la utilizada en las pruebas más largas de la Sección 4.4.

[2] Notar que para pruebas anteriores la convergencia era evidente.

Conclusiones

El trabajo presentado, como se vio en la sección 2.1, tenía por objetivo el desarrollo de un algoritmo de estrategia óptima para el despliegue de niveles de ejercicios, buscando maximizar el rendimiento promedio del estudiante. Este objetivo se cubrió de manera básica abordando un problema introductorio en comparación a la dificultad real práctica del área. También como objetivo secundario surgió la necesidad de crear un simulador de estudiantes para realizar las pruebas, con lo cual se realizó una función que incorporara un comportamiento simple pero suficiente para las pruebas realizadas. Esto deja la posibilidad de que a futuro se mejore este simulador, incorporando más datos, arreglando los saltos en probabilidad producidos al tener una respuesta incorrecta, entre otras cosas; o directamente contar con un grupo de control compuesto de estudiantes con los cuales hacer las pruebas.

Nuevamente es importante recordar que las funciones de valor son un instrumento intermedio para calcular una táctica óptima. También es útil recordar que teóricamente el método de Programación Dinámica, a pesar de encontrar óptimos globales, es muy ineficiente porque su complejidad crece de manera exponencial (ver [4]) a medida que se consideran más estados y/o conexiones entre estados (las acciones que se pueden tomar).

En este trabajo se observó empíricamente lo descrito en el párrafo anterior, específicamente cuando se consideró un error relativo más alto al cambiar un factor de la recompensa (en las primeras pruebas se usó un error nulo con la iteración de las tácticas encontradas). Por otro lado, Monte Carlo es un método de aproximación de las soluciones, que también se ve afectado por el tamaño del problema y su complejidad, además de que juega un rol crucial el parámetro *periodo* para controlar el nivel de precisión con que se obtendrán los promedios para aproximar las funciones de valor de las simulaciones. En relación a lo anterior, se pueden realizar diversas medidas de contención respecto al tiempo, para encontrar una buena aproximación sin sacrificar muchos recursos, como por ejemplo evaluar para determinado problema un *periodo* que permita encontrar buenos promedios para calcular los V; o relajar el problema, considerando una holgura en el error, obteniendo valores de V notoriamente más bajos, pero que se encuentre las mismas tácticas (o similares) que las que se obtendrían en el caso de que no se utilicen medidas de contención como las antes mencionadas. De la mano con estas medidas de contención, está el planteamiento de considerar subdividir el problema, comentado en la sección de análisis, teniendo así muchos subproblemas pero de menor complejidad y que se permita escoger tácticas de manera local, que resulten ser óptimas para cada subproblema.

Por lo anterior surge también la propuesta a considerar una mezcla de ambos métodos,

utilizando las ventajas que tiene cada uno en distintos escenarios. Como por ejemplo sería, el comenzar con una táctica obtenida de PD en un escenario simplificado o más estable en los parámetros iniciales, y a medida que se avanza en el flujo de ejercicios o de sus estados relacionados, con lo cual se adquiere más variabilidad en los datos y parámetros, incorporar una actualización de la táctica a utilizar a partir de MC, dando paso a que convenientemente se puedan ajustar nuevas condiciones de borde, topes, y *periodo*, para que no salga tan costoso en tiempo.

Como este trabajo demuestra, en el cual se analizaron los métodos de Programación Dinámica y Monte Carlo que aborda el caso de dependencia lineal analizado, presenta elementos complejos que requieren ser tratados con cuidado y rigurosidad, es por esto que resulta interesante y desafiante el problema de ampliar las restricciones bajo las que se modelan, para considerar elementos que dan cuenta de un aprendizaje más general y realista. De esta manera, también se podría encontrar una diferencia visual notoria entre los modelos dependientes de 1 estado como los dependientes de 2 estados, puesto que en función por lo que se vio en el presente trabajo, dichos modelos presentaron comportamientos similares en sus resultados (independiente si se usó PD o MC).

En el trabajo expuesto, para tener un problema más simple y abordable en una primera instancia, es que se decidió considerar una progresión lineal en los niveles de ejercicio considerando una sola línea de conocimiento, esto es una versión simplificada e idealizada de lo que se podría encontrar al modelar un proceso de aprendizaje realista, donde la progresión del proceso de aprendizaje resulta ser multidireccional que se entrelaza y bifurca, formando así un grafo donde la interdependencia de los contenidos, ejercicios, niveles y entre otros elementos resulta compleja.

Continuando con la idea de posible trabajo futuro, también está el considerar el tiempo que un estudiante se demore en responder un ejercicio, incorporar gradación en las respuestas (es decir, considerar diferencia entre distintos tipos de respuestas incorrectas), así como considerar la ampliación a la extensión propuesta en este trabajo, considerando tácticas que dependan de más estados del pasado.

Si bien no se contó con datos reales, ni estudiantes de prueba que permitieran corroborar en la práctica la utilidad de los algoritmos estudiados, este trabajo espera ser un primer paso para futuros estudios que apunten a la introducción y desarrollo de metodologías de aprendizaje reforzado en la era digital, que permitan explotar del mejor modo posible las posibilidades tecnológicas del mundo moderno, esto sin dejar de lado que es necesario establecer criterios bien definidos que permitan verificar el nivel de aprendizaje y la efectividad que se obtiene a partir de la introducción de nuevas herramientas metodológicas en el aula de clases.

Bibliografía

[1] K. Binder and D. Heermann. *Monte Carlo Simulation in Statistical Physics*. Springer-Verlag Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-03163-2.

[2] V. Borkar, V. Ejov, J. Filar, and G. Nguyen. *Markov Decision Processes*, pages 49–66. Springer New York, New York, NY, 2012. doi: 10.1007/978-1-4614-3232-6_4.

[3] H. Chang, J. Hu, M. Fu, and S. Marcus. *Simulation-Based Algorithms for Markov Decision Processes*. Springer, 01 2013. doi: 10.1007/978-1-4471-5022-0.

[4] C. Chow and J. Tsitsiklis. The complexity of dynamic programming. *Journal of Complexity*, 5(4):466 – 488, 1989. doi.org/10.1016/0885-064X(89)90021-6.

[5] K. Ciesielski. On Stefan Banach and some of his results. *Banach J. Math. Anal.*, 1(1):6, 2007. doi: 10.15352/bjma/1240321550.

[6] S. Dreyfus. Richard Bellman on the birth of Dynamic Programming. *Operations Research*, 50(1):48–51, 2002. doi: 10.1287/opre.50.1.48.17791.

[7] W. Dunn and K. Shultis. 1 - introduction. In K. Shultis W. Dunn, editor, *Exploring Monte Carlo Methods*, pages 1 – 20. Elsevier, Amsterdam, 2012. doi: 10.1016/B978-0-444-51575-9.00001-4.

[8] R. Feldman and C. Valdez-Flores. *Applied Probability and Stochastic Processes*, chapter 12. Springer, 01 2010. doi: 10.1007/978-3-642-05158-6.

[9] Garychl. Applications of reinforcement learning in real world. `https://bit.ly/3f63BHq`, Agosto 2018. [Web; accedido el 30-07-2020].

[10] K. Jones. How technology is shaping the future of education. `https://www.visualcapitalist.com/how-technology-is-shaping-the-future-of-education`, Febrero 2020. [Web; accedido el 30-07-2020].

[11] L. Kaelbling, M. Littman, and A. Moore. Reinforcement Learning: A Survey. *J. Artif. Int. Res.*, 4(1):237–285, May 1996. doi: 10.1613/jair.301.

[12] L. Kallenberg. *Markov Decision Processes - version 2016*. University of Leiden, 2016.

[13] A. Lazaric. Markov decision processes and dynamic programming. `http://chercheurs.lille.inria.fr/~lazaric/Webpage/MVA-RL_Course13.html`, Octubre 2013. [Web;

accedido el 04-02-2020].

[14] Y. Li. Reinforcement Learning Applications. *ArXiv*, abs/1908.06973, 2019. Preprint.

[15] R. Raja and P. Nagasubramani. Impact of modern technology in education. *Journal of Applied and Advanced Research*, 3:33, 05 2018. doi: 10.21839/jaar.2018.v3iS1.165.

[16] T. Scott. *Better Education Through Improved Reinforcement Learning*. PhD thesis, University of Washington, 2017.

[17] B. Settles. How we learn how you learn. `https://making.duolingo.com/how-we-learn-how-you-learn`, Diciembre 2016. [Web; accedido el 30-07-2020].

[18] P. Songmuang and M. Ueno. E-testing construction support system with some prediction tools. In Mary Beth Rosson, editor, *Advances in Learning Processes*, chapter 4. IntechOpen, Rijeka, 2010. doi: 10.5772/7942.

[19] C. Soto, X. Saavedra, I. Larraguibel, and F. Flores. Estadísticas de la educación 2016. *Centro de Estudios MINEDUC, División de Planificación y Presupuesto*, 2017.

[20] R. Sutton and A. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[21] F. Tobar. *Apunde de curso MA5204-1 (Aprendizaje de Máquinas)*. DIM/CMM, FCFM, Universidad de Chile, Otoño 2018.