1era Ed.
2024
Vol. 1

PUERTO MADERO
EDITORIAL

FUNDAMENTOS DE LA INTELIGENCIA ARTIFICIAL:
UNA VISION INTRODUCTORIA

PATRICIO XAVIER MORENO VALLEJO
GISEL KATERINE BASTIDAS GUACHO
PATRICIO RENE MORENO COSTALES

puertomaderoeditorial.com.ar
La Plata - Argentina

FUNDAMENTOS DE LA INTELIGENCIA ARTIFICIAL: UNA VISION INTRODUCTORIA

Volumen I

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

FUNDAMENTOS DE LA INTELIGENCIA ARTIFICIAL: UNA VISION INTRODUCTORIA

Volumen I

AUTORES:
PATRICIO XAVIER MORENO VALLEJO
GISEL KATERINE BASTIDAS GUACHO
PATRICIO RENE MORENO COSTALES

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General

Moreno Vallejo, Patricio Xavier
Fundamentos de la inteligencia artificial : una visión introductoria / Patricio Xavier Moreno Vallejo ; Gisel Katerine Bastidas Guacho ; Patricio Rene Moreno Costales ; Editado por Juan Carlos Santillán Lima ; Daniela Margoth Caichug Rivera. - 1a ed revisada. - La Plata : Puerto Madero Editorial Académica, 2024.
Libro digital, PDF

Archivo Digital: descarga y online
ISBN 978-631-6557-23-0

1. Ingeniería Informática. 2. Inteligencia Artificial. I. Bastidas Guacho, Gisel Katerine. II. Moreno Costales, Patricio Rene. III. Santillán Lima, Juan Carlos, ed. IV. Caichug Rivera, Daniela Margoth, ed. V. Título.
CDD 006.301

ISBN Tomo 1

Moreno Vallejo, Patricio Xavier
Fundamentos de la inteligencia artificial : una visión introductoria / Patricio Xavier Moreno Vallejo ; Gisel Katerine Bastidas Guacho ; Patricio Rene Moreno Costales ; Editado por Juan Carlos Santillán Lima ; Daniela Margoth Caichug Rivera. - 1a ed - La Plata : Puerto Madero Editorial Académica, 2024.
Libro digital, PDF

Archivo Digital: descarga y online
ISBN 978-631-6557-25-4

1. Aplicaciones Informáticas. 2. Inteligencia Artificial. I. Bastidas Guacho, Gisel Katerine. II. Moreno Costales, Patricio Rene. III. Santillán Lima, Juan Carlos, ed. IV. Caichug Rivera, Daniela Margoth, ed. V. Título.
CDD 006.301

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

DEDICATORIA

A los niñ@s: Leonel Robayo Moreno, Victoria Robayo Moreno y Benjamín Moreno Bastidas.

A quienes inspiraron esta obra: Peter Norving, Stuart Russell, Brian Yu y Hugo Banda.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Primera Edición, Julio 2024

FUNDAMENTOS DE LA INTELIGENCIA ARTIFICIAL: UNA VISION INTRODUCTORIA
ISBN General: 978-631-6557-23-0
ISBN Tomo 1: 978-631-6557-25-4
ISBN Tomo 2: 978-631-6557-26-1

Editado por:
Sello editorial: ©Puerto Madero Editorial Académica
Nº de Alta: 933832

Editorial: © Puerto Madero Editorial Académica
CUIL: 20630333971
Calle 45 N491 entre 4 y 5
Dirección de Publicaciones Científicas Puerto Madero Editorial Académica
La Plata, Buenos Aires, Argentina
Teléfono: +54 9 221 314 5902
+54 9 221 531 5142
Código Postal: AR1900

Este libro se sometió a arbitraje bajo el sistema de doble ciego (peer review)

Corrección y diseño:
Puerto Madero Editorial Académica
Diseñador Gráfico: José Luis Santillán Lima

Diseño, Montaje y Producción Editorial:
Puerto Madero Editorial Académica
Diseñador Gráfico: Santillán Lima, José Luis

Director del equipo editorial: Santillán Lima, Juan Carlos

Editor: Santillán Lima, Juan Carlos
Caichug Rivera, Daniela Margoth

Hecho en Argentina
Made in Argentina

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

AUTORES:

Patricio Xavier Moreno Vallejo

Escuela Superior Politécnica de Chimborazo, Facultad de Administración de Empresas, Carrera de Gestión del Transporte, Panamericana Sur Km 1 1/2, EC060155, Riobamba, Chimborazo, Ecuador

pxmoreno@espoch.edu.ec

https://orcid.org/0000-0002-9317-9884

Gisel Katerine Bastidas Guacho

Escuela Superior Politécnica de Chimborazo, Facultad de Informática y Electrónica, Carrera de Software, Panamericana Sur Km 1 1/2, EC060155, Riobamba, Chimborazo, Ecuador

gis.bastidas@espoch.edu.ec

https://orcid.org/0000-0002-6070-7193

Patricio René Moreno Costales

Escuela Superior Politécnica de Chimborazo, Facultad de Informática y Electrónica, Carrera de Software, Panamericana Sur Km 1 1/2, EC060155, Riobamba, Chimborazo, Ecuador

pmoreno@espoch.edu.ec

https://orcid.org/0000-0001-8255-8953

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

CONTENIDO

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ÍNDICE DE FIGURAS

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ÍNDICE DE TABLAS

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

RESUMEN

El presente estudio sobre inteligencia artificial inicia con la cronológica histórica de la inteligencia artificial, cómo esta ciencia contribuye e influye en diversas áreas en la actualidad, se analiza el diseño de los agentes inteligentes, las propiedades de sus entornos de trabajo, la forma en que adquieren el conocimiento y lo representan los humanos, para poder ofrecer un formato accesible y utilizable por sistemas de inteligencia artificial.

La lógica es fundamental en la inteligencia artificial para la representación del conocimiento a través de la lógica proposicional y de primer orden. Mediante la inferencia se realiza el razonamiento lógico y la toma de decisiones en sistemas inteligentes. Las reglas de producción que son expresiones lógicas permiten representar las bases de conocimientos.

Se examinan varios métodos y estrategias de búsqueda no informada e informada, con el propósito de seleccionar la solución más adecuada de entre un conjunto de alternativas posibles se analiza los procesos de optimización. Los árboles de decisión al ser ampliamente utilizados se revisa su estructura y el modelo empleado en su evaluación.

Palabras clave: Inteligencia Artificial, Conocimiento, Agentes, Búsquedas

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

INTRODUCCIÓN

La inteligencia artificial se encarga de representar las diferentes formas de conocimiento que posee el ser humano. Sus herramientas son la abstracción y la inferencia que le permiten crear sistemas de computación no convencionales los cuales son capaces de aprender nuevas tareas en base a la información proporcionada y tomar decisiones en entornos diversos en los cuales debe actuar.

La base de la inteligencia artificial se encuentra en el influyente artículo de Alan Turing, "Computing Machinery and Intelligence", el cual sostiene que las máquinas deben considerarse inteligentes si pueden comportarse como humanos.

Este libro, está dividido en cuatro capítulos, en los cuales se explora los fundamentos y usos de la inteligencia artificial. El primer capítulo profundiza en los conceptos básicos de la inteligencia artificial. Analiza su evolución desde la década de 1950 y se examina cómo otras disciplinas han ayudado para su desarrollo; tales como las matemáticas, filosofía, psicología, lingüística y ciencia de la computación. Se destaca el impacto de la inteligencia artificial en numerosos sectores como la ciencia, ingeniería, empresas, medicina y planificación estratégica.

Además, se trata sobre el concepto de agentes inteligentes, se explora sus diseños, analizando su efectividad, identificando varias arquitecturas de programas para agentes. Se busca resolver problemas mediante el razonamiento basado en el conocimiento y su representación desde la perspectiva de la inteligencia simbólica.

La lógica es tratada en el capítulo II, constituye una técnica para la representación del conocimiento utilizando proposiciones y operadores lógicos. Mediante las cláusulas de Horn se representa el conocimiento, del cual, se parte para poder razonar y luego ejecutar decisiones inteligentes.

El Capítulo III, está relacionado a las búsquedas, que constituyen estrategias esenciales en la inteligencia artificial para alcanzar metas específicas. Se explora tanto las formas de búsquedas no informadas como las informadas, destacando sus diferencias y aplicaciones. También se aborda los conceptos de optimización y árboles de decisión.

Este libro está diseñado para ofrecerles a ustedes estimados lectores una sólida comprensión de los fundamentos de la inteligencia artificial. Nuestro objetivo es animarlos a que indaguen el potencial que posee la inteligencia artificial y contribuir a su desarrollo.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

CAPITULO 1

1 EXPLORANDO LA INTELIGENCIA ARTIFICIAL

A la pregunta ¿qué es la inteligencia? la respuesta constituye que es la capacidad de establecer relaciones, las cuales se manifiestan en los seres humanos a través del pensamiento y la parte intelectual, y en los animales de manera puramente sensorial por medio de los sentidos (Artasanchez & Joshi, 2020). Los humanos tenemos múltiples aspectos que son parte de la inteligencia, como la capacidad de comunicación, aprendizaje, razonamiento abstracto y resolución de problemas, mientras en los animales la inteligencia está relacionado a poderse adaptar al medio ambiente, manifestándose como la capacidad que poseen de la percepción, memoria y toma de decisiones. Entre los humanos y los animales compartimos rasgos básicos de inteligencia como la conciencia del entorno y la habilidad para responder a estímulos (McCarthy, 1956).

La reflexión simbólica y de autorreflexión nos permite trascender el mundo tangible y sensorial, crear cultura y tecnología, ganar experiencia y entender conceptualmente el mundo que nos rodea de forma objetiva. La aplicación del conocimiento es útil en el análisis y resolución de problemas, en el razonamiento durante la toma de decisiones; fomenta la predisposición al descubrimiento, invención, creatividad e innovación. La inteligencia permite a las personas afrontar situaciones complejas de forma oportuna, rápida y razonada, y tener la capacidad de predecir el desarrollo y los cambios en el entorno que nos rodea.

El año 1950 marca el inicio formal de la inteligencia artificial, debido a que fue entonces cuando se buscó dotar a las máquinas de recursos para resolver problemas de manera autónoma, sin depender del apoyo humano.

Si bien la noción de máquina ha estado presente durante muchos años, fue Alan Turing quien sentó las bases de la inteligencia artificial a finales de la Segunda Guerra Mundial, en particular al estudiar científicamente las máquinas inteligentes. Turing había estado trabajando en la teoría de la computabilidad desde la década de 1930 y desarrolló una de las primeras computadoras electromecánicas, se le recuerda por haber desarrollado una máquina para descifrar códigos llamada "bombe" en Bletchley Park, Reino Unido, que descifraba mensajes encriptados con la máquina alemana Enigma. En octubre de 1950, mientras trabajaba en la Universidad de Manchester publicó su artículo "Computing Machinery and Intelligence" en la revista Mind (Oxford University Press), donde

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

introdujo la prueba de Turing, el aprendizaje autónomo, los algoritmos genéticos y el aprendizaje por refuerzo, sentando las bases conceptuales para que los científicos comenzaran a explorar cómo crear programas capaces de resolver problemas ellos mismos, sin la intervención humana. Este artículo es considerado el punto de partida que impulsó la investigación formal en inteligencia artificial, dando origen a una nueva disciplina dedicada a replicar la inteligencia humana en las máquinas. El legado de Turing ha inspirado décadas de avances en aprendizaje automático, procesamiento de lenguaje natural y otros campos clave de la IA.

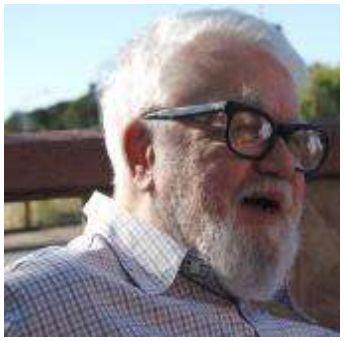


Figura 1.1 John McCarthy

Fuente: Wikipedia

En 1956, el informático John McCarthy (1927-2011) acuñó el término "Inteligencia Artificial" y organizó el influyente proyecto de investigación Dartmouth Summer Research Project on Artificial Intelligence, en el que participaron varios científicos estadounidenses de matemáticas, psicología, ciencias de la computación y teoría de la información como Marvin Minsky, Allen Newell, Arthur Samuel y Herbert Simon. En aquella época, las computadoras se utilizaban para tareas aritméticas con herramientas de programación muy primitivas. El entusiasmo y esfuerzo pionero de McCarthy y sus colegas en Dartmouth sentaron las bases conceptuales y metodológicas para el nacimiento de la IA como campo formal. Su trabajo en esas primitivas condiciones informáticas es digno de reconocimiento por impulsar una disciplina que décadas después tendría aplicaciones transformadoras. McCarthy se refiere a esa época como la era de "¡Mira mamá, ahora sin manos!".

Desde sus inicios, la Inteligencia Artificial (IA) ha tenido como objetivo replicar y ampliar las capacidades humanas mediante la imitación y expansión de la inteligencia humana mediante medios y técnicas artificiales. Para lograr este propósito, la IA se ha centrado en el desarrollo de modelos computacionales que abordan el comportamiento inteligente. Su enfoque se centra en la creación de sistemas informáticos capaces de realizar actividades como la percepción, el razonamiento, el aprendizaje, la asociación, la toma de decisiones y la resolución de problemas complejos.

El progreso que se ha alcanzado en la actualidad en IA ha permitido automatizar actividades que antes solo los expertos humanos podían realizar debido a su complejidad

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

o necesidad del juicio de un experto, la IA ha redefiniendo lo que entendemos por procesos cognitivos e inteligentes, hiendo más allá de las restricciones inherentes a la biología humana (Whitby, 2009).

La Inteligencia Artificial representa una disciplina dentro de la informática que busca construir máquinas capaces de funcionar de manera autónoma en entornos complejos y en constante cambio, creando sistemas que sean capaces de adaptarse y responder de manera efectiva a situaciones desafiantes.

1.1 Breve historia cronológica de la inteligencia artificial

En 1943, Warren McCulloch y Walter Pitts presentaron el primer modelo de red neuronal artificial capaz de aprender y resolver funciones lógicas. Durante la década de 1950, la investigación en inteligencia artificial se enfocó principalmente en los juegos. En 1951, Minsky y Edmonds construyeron el primer computador basado en una red neuronal llamado SNARC. En 1956, Arthur Samuel desarrolló el primer programa heurístico de juego con capacidad de aprendizaje. Ese mismo año, Allen Newell, Herbert Simon y Cliff Shaw inventaron el programa heurístico llamado Logic Theorist, que resolvió correctamente 38 de los primeros 52 teoremas del Principia Mathematica. Este trabajo marcó el inicio de la investigación en psicología cognitiva utilizando ordenadores.

En 1957, Noam Chomsky presentó su obra "Estructuras Sintácticas", destacando la importancia de la sintaxis en la investigación del lenguaje formal (Chomsky, 2004). En 1958, John McCarthy inventó el lenguaje Lisp, una herramienta utilizada para la investigación en inteligencia artificial que podía procesar tanto valores numéricos como símbolos (McCarthy, 1960). McCarthy también fue pionero en el desarrollo del concepto de tiempo compartido (Barski, 2010).

A principios de la década de 1960, la investigación en inteligencia artificial se centró principalmente en algoritmos de búsqueda y resolución de problemas generales. Allen Newell, Herbert Simon y Cliff Shaw desarrollaron un programa llamado GPS (General Problem Solver), que era un sistema definido por objetos y operadores que se aplicaban a dichos objetos. GPS pudo resolver problemas como las Torres de Hanoi. El sistema utilizaba reglas heurísticas para aprender a partir de sus propios descubrimientos y seguía un enfoque similar al de los humanos para resolver problemas, estableciendo un plan, aplicando axiomas y reglas, y analizando medios y fines para modificar la resolución del problema hasta alcanzar el objetivo.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

En 1961, Marvin Minsky publicó el artículo "Pasos hacia la Inteligencia Artificial", estableciendo una terminología unificada para la investigación en inteligencia artificial. En 1962, Frank Rosenblatt desarrolló el Perceptrón, una red neuronal utilizada para el reconocimiento visual de patrones, aunque tuvo un impacto limitado en ese momento.

En 1965, Edward Feigenbaum, Bruce Buchanan, Joshua Lederberg y Carl Djerassi desarrollaron el primer sistema experto llamado DENDRAL, utilizado para el análisis químico. Ese mismo año, Alan Robinson propuso el principio de la Resolución. En 1968, Ross Quillian introdujo la Red Semántica como una representación del conocimiento. En 1969, se fundó la Conferencia Conjunta Internacional de Inteligencia Artificial (IJCAI), que comenzó a celebrarse cada dos años y, a partir de 2016, se lleva a cabo anualmente. La revista Intelligence Artificial se publica bajo los auspicios de la IJCAI desde 1970.

Después de la década de 1970, cuando las redes neuronales no cumplieron sus promesas, época conocida como el "AI hype", la financiación y las actividades de investigación se redujeron drásticamente. Esto se conocio como el "invierno de la IA". A principios de la década de 1970, el campo de la inteligencia artificial se enfocó hacia el estudio y la comprensión del lenguaje natural, así como a la representación avanzada del conocimiento. Durante este período, se observaron progresos significativos en diversas ramas de la IA, marcando un hito en su desarrollo.

En 1972, un avance fundamental fue la publicación de Terry Winograd sobre el programa SHRDLU. Este programa se distinguió por su habilidad pionera en comprender el lenguaje natural, específicamente el inglés, demostrando una comprensión profunda y contextual de las interacciones lingüísticas. SHRDLU no solo representó un logro técnico importante, sino que también estableció un marco conceptual crítico para las investigaciones futuras en la comprensión del lenguaje dentro del campo de la inteligencia artificial. Alain Colmerauer, en la Universidad de Marsella en Francia, desarrolló el lenguaje de programación Prolog en el mismo año. Prolog se convirtió en un lenguaje ampliamente utilizado en la programación de IA debido a su capacidad para trabajar con la lógica y el razonamiento (Colmerauer & Roussel, 1996).

En 1973, Roger Schank propuso la teoría de la dependencia conceptual, que se centraba en la comprensión del lenguaje natural basada en el conocimiento y la

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

representación semántica. Esta teoría contribuyó al avance en el procesamiento del lenguaje natural en la IA.

Marvin Minsky publicó en 1974 la teoría del sistema de marco (A Framework for Representing Knowledge), que fue una contribución importante en la representación del conocimiento en la IA (Baader, 2003). Esta teoría posibilitó que las máquinas ordenaran y estructuraran la información de una forma parecida a como lo hacen los seres humanos, lo que simplificó el proceso de entendimiento y razonamiento (Minsky & others, 1974).

En ese mismo año, Edward Shortliffe desarrolló el sistema experto MYCIN como parte de su tesis doctoral, implementado en Lisp, era capaz de diagnosticar trastornos en la sangre y prescribir medicación correspondiente. Este sistema experto fue un hito en el campo médico y demostró el potencial de la IA en el diagnóstico y tratamiento de enfermedades (Shortliffe, 1977). Durante esta época, también se desarrollaron sistemas expertos más avanzados, como el EURISKO, que tenía la capacidad de mejorar automáticamente su conjunto de reglas heurísticas mediante la inducción.

Además, en los años 70 se lograron otros avances notables, como la popularización de los algoritmos genéticos gracias al trabajo de John Holland, y la resurrección de las redes neuronales con la creación del algoritmo de retropropagación (backpropagation) por parte de Paul John Werbos. Este algoritmo sigue siendo ampliamente utilizado en el aprendizaje supervisado para el entrenamiento de redes neuronales (Pham & Karaboga, 2012). Asimismo, en esta década se construyó el primer robot capaz de comprender el inglés y se desarrolló el primer vehículo autónomo, marcando importantes avances en la interacción entre la IA y el mundo físico.

En 1977, Edward Feigenbaum publicó un paper titulado "El arte de la inteligencia artificial: Temas y estudios de caso en ingeniería del conocimiento" en el quinto IJCAI. En esta publicación dice que la ingeniería del conocimiento es el arte de aplicar los principios y herramientas de la investigación en IA para abordar problemas de aplicaciones difíciles que requieren conocimientos especializados para su solución (Brachman & Levesque, 2004).

En 1978, John P. McDermott introdujo un hito en el campo de la inteligencia artificial con el desarrollo del sistema experto R1, marcando uno de los primeros éxitos significativos en esta área. Compuesto por aproximadamente 2500 reglas, R1 fue implementado por DEC (Digital Equipment Corporation) para optimizar los procesos de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

pedidos de clientes, facilitando la selección de componentes adecuados para sistemas informáticos. Este logro no solo demostró la utilidad práctica de los sistemas expertos, sino que también abrió camino para su aplicación en el mundo empresarial.

Posteriormente, a mediados de la década de 1980, se produjeron avances importantes en el algoritmo de aprendizaje por retroalimentación, originalmente desarrollado por Bryson y Ho en 1969. Estas mejoras fueron importantes para el progreso de los métodos de aprendizaje automático, potenciando significativamente el rendimiento y la eficiencia en el entrenamiento de sistemas basados en IA, lo que representó un avance clave en la evolución de las técnicas de inteligencia artificial.

Durante los años 80, la Inteligencia Artificial se consolida como una ciencia en sí misma. Judea Pearl en 1982 promueve la idea de los sistemas expertos normativos que se basan en normas y principios de racionalidad formal, los cuales actúan racionalmente de acuerdo con las leyes de la teoría de la decisión, sin intentar replicar directamente el proceso de razonamiento humano. En 1986, Eric Horvitz y David Heckerman que han realizado contribuciones significativas en áreas como el aprendizaje automático, la toma de decisiones basada en datos y la bioinformática, respaldaron la idea de Pearl.

En ese mismo año, Rumelhart y McClelland publicaron aplicaciones de algoritmos de aprendizaje en informática y psicología, lo que dio lugar al establecimiento del enfoque del conexionismo basado en redes neuronales. Este enfoque revivió el interés en las redes neuronales como una herramienta poderosa para el aprendizaje automático y estableció una nueva dirección en la investigación y desarrollo de IA.

En 1988, Judea Pearl publica el influyente texto "Probabilistic Reasoning in Intelligent Systems", que incorpora conceptos de probabilidades y teoría de decisiones en el campo de la inteligencia artificial. Este trabajo fue fundamental para avanzar en el uso de modelos probabilísticos en la toma de decisiones en sistemas inteligentes.

En 1995, Russell y Norvig retomaron el trabajo en agentes totales, sistemas que buscan encontrar soluciones a partir de información de entrada continua. En 1997, la computadora Deep Blue de IBM venció al ajedrecista ruso Garry Kasparov considerado uno de los mejores jugadores de ajedrez, este hecho situó a la inteligencia artificial como protagonista en el ámbito tecnológico y demostró el potencial que tiene al desafiar a expertos humanos en juegos complejos.

En el año 2000, se construyó el robot ASIMO, que era capaz de desplazarse en

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

dos pies, dar la mano y contestar a preguntas simples. En 2002, la empresa Cognitec desarrolla el primer sistema comercial de reconocimiento facial a partir de imágenes o videos. En 2006, se presentó el prototipo del robot programable y autónomo NAO. En 2009, se lograron desarrollar robots sociales que son capaces de detectar emociones e interactuar con niños autistas.

A partir del 2010 hasta la actualidad, se ha desarrollado el aprendizaje profundo (DL Deep Learning), que es un tipo especial de red neuronal que tiene más de una capa oculta. Esto solo es posible con el aumento de la potencia informática, especialmente de las unidades de procesamiento gráfico (GPU) y algoritmos mejorados. Hasta ahora, el DL ha superado muchos otros algoritmos en un gran conjunto de datos.

En 2014, se produjo un acontecimiento importante en el campo de la inteligencia artificial cuando el chatbot Eugene Goostman logró superar el test de Turing, considerado un indicador crítico en la evaluación de la capacidad de las máquinas para imitar la inteligencia humana en el diálogo. Este logro no solo demostró avances significativos en el desarrollo de sistemas de conversación automatizados, sino que también abrió un debate sobre la complejidad y la verosimilitud de las interacciones entre humanos y máquinas, marcando un punto de referencia en la historia de la inteligencia artificial.

En el año 2016, AlphaGo, un programa de IA creado por DeepMind Technologies, alcanzó notoriedad mundial al vencer a Lee Sedol, un campeón global en Go, un milenario y desafiante juego de estrategia chino caracterizado por su alta complejidad y las numerosas posibilidades que ofrece cada partida. Este logro marcó un punto de inflexión, generando un renovado interés y avances significativos en el ámbito de la inteligencia artificial, evidenciando así su capacidad para abordar tareas de gran complejidad y para la toma de decisiones estratégicas.

Un hito importante en la intersección de la inteligencia artificial y la biología se alcanzó en 2020 gracias a DeepMind, los creadores de AlphaGo. Desarrollaron AlphaFold, una versión mejorada de su sistema de IA, que logró descifrar un desafío que había perplejado a los biólogos por más de cinco décadas: el plegamiento de proteínas. Este fenómeno, en el que las proteínas obtienen su forma tridimensional específica, es importante para comprender su función dentro del organismo. Este avance es fundamental, debido a que desentraña cómo las células operan y cómo se manifiestan diversas enfermedades a nivel molecular, debido a la extraordinaria complejidad de este

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

proceso.

A partir del 2021, los avances en inteligencia artificial han sido impresionantes, transformando diversos sectores de manera significativa. Un claro ejemplo es el modelo GPT de OpenAI, que ha marcado un antes y un después en la forma en que las máquinas entienden y generan lenguaje, siendo útil en una variedad de aplicaciones que incluyen desde la generación automatizada de contenido hasta soporte en servicios al cliente. Por otro lado, los modelos como BERT, desarrollados por Google, han revolucionado la manera en que se procesa el lenguaje en búsquedas en línea y otras aplicaciones, logrando una mayor precisión y relevancia en las respuestas brindadas. IBM Watson Health ha transformado el sector médico con su capacidad para analizar extensos volúmenes de datos médicos, contribuyendo a diagnósticos más acertados y tratamientos personalizados.

La empresa Boston Dynamics ha sobresalido con sus robots Atlas y Spot, que se han convertido en herramientas importantes para tareas de inspección y rescate, así como en aplicaciones industriales, demostrando la evolución y adaptabilidad de la robótica. Hugging Face, ha permitido la accesibilidad al aprendizaje profundo y procesamiento del lenguaje natural a un público más amplio. En el terreno creativo, proyectos como AIVA (Artificial Intelligence Virtual Artist) están trabajando en la composición musical mediante la inteligencia artificial y DeepArt explora nuevas posibilidades en el arte visual a través de la tecnología de IA. Google AI for Earth está utilizando la inteligencia artificial para el análisis de datos medioambientales, en la búsqueda de soluciones sostenibles, que aborden los retos más críticos del cambio climático.

El campo de la inteligencia artificial se suele confundir con la ciencia de datos, el big data y la minería de datos, pero en realidad representan disciplinas diferentes. La ciencia de datos se enfoca principalmente en la manipulación y el análisis de datos, incluyendo el manejo de grandes volúmenes de datos (big data) y la minería de datos; de forma frecuente emplea técnicas de aprendizaje automático (Machine Learning) y aprendizaje profundo (Deep Learing) para el procesamiento de datos.

La transformación digital ha generado una inmensa cantidad de datos, los cuales provienen de diversas fuentes, al no poder ser analizados de manera eficiente con las herramientas y algoritmos tradicionales de bases de datos; la inteligencia artificial, el big data y la minería de datos han desarrollo algoritmos sofisticados (algoritmos de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

aprendizaje automático, algoritmos de clasificación y regresión, algoritmos de agrupamiento y algoritmos de minería de datos), que posibilitan a las empresas tomar decisiones basadas en datos y brindar a los usuarios acceso a información relevante de manera rápida y cómoda en sus actividades diarias.

Google Search y Google Maps proporcionan resultados de búsqueda relevantes y sugerencias de navegación personalizadas comparado con productos similares, Spotify recomienda música que se ajuste a los gustos y preferencias del usuario y Netflix emplea la IA para sugerir películas y series que probablemente interesen al usuario, basándose en sus visualizaciones anteriores y preferencias. Todas estas plataformas populares incorporan algoritmos avanzados de inteligencia artificial para realizar su trabajo.

Al usar uno de los siguientes asistentes personales, Alexa de Amazon, Siri de iPhone, Cortana de Microsoft, Bixby de Samsung o Google Assistant, observamos que entiende lo que hablamos, ejecutando de forma eficiente la tarea solicitada. Las redes sociales como Facebook, Pinterest y Google Fotos, tienen incorporado el reconocimiento facial, a través de técnicas sofisticadas de visión por computadora que mejora la interactividad y seguridad de sus usuarios.

En el ámbito empresarial, la robótica, los sistemas expertos y el aprendizaje automático están transformando los procesos y brindando oportunidades de innovación en los negocios.

Actualmente, se están desarrollando herramientas y plataformas de ciencia de datos que permiten la aplicación de algoritmos de inteligencia artificial. Algunos ejemplos de estas plataformas incluyen RapidMiner, Anaconda y Python.

Los líderes de la industria tecnológica, como Google, Microsoft, Meta Platforms, Amazon, Oracle y OpenAI están profundamente comprometidos con la inteligencia artificial. Esta tecnología se está integrando profundamente en diversos aspectos de nuestra vida cotidiana, influyendo en nuestras decisiones y siendo importante en las estrategias empresariales, la salud pública, la seguridad y las finanzas. En el futuro debemos esperar un desarrollo significativo en aplicaciones web, videojuegos y robótica autónoma, veremos cada vez más vehículos autónomos circulando por las calles, robots sociales ayudando a las personas en su vida diaria y exploradores planetarios llegando a lugares más lejanos en el universo. Asimismo, las aplicaciones en el ámbito medioambiental y el ahorro energético serán de gran importancia, al igual que en los

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

campos de la economía, la sociología y el arte (Lee & Qiufan, 2021).

1.2 Contribuciones de las ciencias a la inteligencia artificial

Varias ciencias han aportado al desarrollo de la Inteligencia Artificial, entre ellas se tiene:

La ***filosofía*** ha planteado diversas teorías sobre el razonamiento y el aprendizaje, concibiendo a la mente como una máquina que funciona a partir del conocimiento, codificado en un lenguaje interno y al considerar que el pensamiento sirve para determinar cuál es la acción correcta que se debe emprender (McCarthy, 1990).

Un parte relevante en el desarrollo del razonamiento deductivo es la lógica silogística propuesta por el filósofo Aristóteles, considerada el primer sistema formal de razonamiento deductivo, la cual establece reglas precisas para derivar conclusiones válidas a partir de premisas, esto a contribuido al desarrollo posterior de la lógica y ha influido en los fundamentos de la inteligencia artificial.

Desde la perspectiva de la inteligencia artificial, se han explorado puntos de vista más avanzados y sofisticados para comprender el razonamiento y el aprendizaje; que involucran el uso de algoritmos de aprendizaje automático, redes neuronales y técnicas de procesamiento de lenguaje natural, permitiendo a las máquinas no solo realizar razonamiento deductivo, sino también abordar problemas complejos de manera más eficiente y aproximarse a la toma de decisiones basadas en datos y modelos de conocimiento.

Las ***matemáticas*** son fundamentales en el desarrollo de teorías formales relacionadas con la lógica, las probabilidades, la teoría de decisiones y la computación. La lógica matemática permite tratar con declaraciones y razonamientos que pueden ser verdaderos o falsos, mientras que la teoría de probabilidades aborda situaciones de incertidumbre y riesgo.

En el libro "Una investigación de las leyes del pensamiento" de George Boole se encuentra el desarrollo del álgebra booleana que se ha convertido en una herramienta esencial en la lógica digital y la computación para la representación de reglas básicas de razonamiento en actividades mentales, permitiendo manipular datos y realizar operaciones lógicas en las computadoras y otros dispositivos electrónicos.

Las matemáticas siguen siendo esenciales en el ámbito de la inteligencia artificial.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Las redes neuronales y los algoritmos de aprendizaje automático son modelos matemáticos que permiten a las máquinas aprender de los datos y realizar inferencias complejas para tomar decisiones o realizar acciones basadas en la información procesada.

La teoría de la computación proporciona el marco teórico y los principios que guían la creación y el funcionamiento de algoritmos que minimicen el uso de recursos, tiempo de computación y memoria; mientras que los fundamentos matemáticos aportan las herramientas y técnicas necesarias para formular y resolver problemas de manera lógica y precisa (Domkin, 2021).

La ***psicología*** tiene gran importancia en el estudio y entendimiento de la mente humana, al ser una ciencia, se enfoca en explorar y comprender los procesos cognitivos (como el pensamiento, la memoria, la percepción y la toma de decisiones) y emocionales (los sentimientos y las emociones) que caracterizan el comportamiento humano.

La idea de que los humanos procesan información ha contribuido a modelar los sistemas de inteligencia artificial, inspirándose en la forma en que los humanos pensamos y tomamos decisiones. Los enfoques teóricos y conceptuales de la psicología sobre el procesamiento del lenguaje natural, la percepción visual y la toma de decisiones, han influido en el diseño de algoritmos y modelos utilizados en los sistemas de la inteligencia artificial.

La IA no se centra exclusivamente en imitar la mente humana. También incorpora elementos de matemáticas, estadística e informática, lo que permite a las máquinas realizar tareas cognitivas específicas y tomar decisiones en diversos contextos, a veces de manera diferente a como lo haría un ser humano. La ***lingüística*** es importante en la inteligencia artificial al proporcionar teorías y herramientas para comprender la estructura y el significado del lenguaje humano. El lenguaje es un componente fundamental de la comunicación y su comprensión y generación son elementos esenciales en muchos sistemas de inteligencia artificial.

La ***lingüística*** es la ciencia del lenguaje que posee un enfoque sistemático para el análisis de la gramática (estructura y reglas del lenguaje), semántica (significado de las palabras y frases) y pragmática (uso del lenguaje en contextos específicos) del lenguaje. El estudio detallado de las reglas y patrones lingüísticos, ha permitido los avances tecnológicos en el campo de la inteligencia artificial y la computación, permitiendo a las máquinas entender y generar lenguaje, así como para producir texto comprensible para

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

los humanos.

La colaboración entre la lingüística y la inteligencia artificial ha sido un factor importante para el progreso del procesamiento del lenguaje natural (PLN), que permite a las máquinas entender, interpretar y generar lenguaje humano de forma similar a cómo lo hacen las personas. Otras aplicaciones son la traducción automática; los chatbots, que pueden interactuar en lenguaje natural con los usuarios; y los sistemas de diálogo, que facilitan una comunicación fluida y natural con las máquinas.

La lingüística ha sido fundamental en la creación de recursos como bases de datos léxicas (listas de palabras y sus significados), corpus textuales (grandes colecciones de textos escritos o transcripciones de habla) y modelos de conocimiento lingüístico. Estos recursos son indispensables para entrenar y mejorar los sistemas de IA, proporcionando la información necesaria para que las máquinas aprendan y entiendan el lenguaje humano.

La **ciencia de la computación** proporciona los lenguajes de programación que son el medio por el cual se escriben y se ejecutan los algoritmos de inteligencia artificial y los marcos de desarrollo, que ofrecen estructuras y librerías predefinidas para facilitar la creación de aplicaciones de inteligencia artificial.

Alan Turing es una figura fundamental en el origen de la ciencia de la computación debido a sus contribuciones innovadoras y su visión de lo que podrían ser las computadoras y su funcionamiento. Sus teorías y modelos proporcionan el marco conceptual para comprender cómo las máquinas pueden no solo procesar información, sino también realizar tareas que requieren algún nivel de inteligencia, como el aprendizaje, el razonamiento y la adaptación.

En el ámbito de la IA, los programas y algoritmos son fundamentales para capacitar a las máquinas para que realicen tareas consideradas inteligentes como la resolución de problemas hasta la imitación de comportamientos humanos como el aprendizaje y la toma de decisiones. Los programas demandan una cantidad significativa de recursos computacionales, incluyendo un alto poder de procesamiento y una gran capacidad de memoria. La eficiencia de un algoritmo se refiere a cuán bien utiliza los recursos disponibles (como tiempo de cómputo y memoria) para lograr sus objetivos

La ciencia de la computación ha avanzado en el procesamiento de datos (capacidad de manejar y transformar datos en información útil), el aprendizaje automático (sistemas capaces de aprender y mejorar a partir de la experiencia), la optimización de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

algoritmos (velocidad y uso de recursos) y la gestión de grandes volúmenes de información (procesar y analizar grandes conjuntos de datos).

La ***neurociencia*** se centra en entender el funcionamiento del cerebro humano a niveles: molecular, celular y de comportamiento. Este estudio permite desarrollar modelos y principios que son aplicados en la creación de inteligencia artificial. También, proporciona información valiosa sobre cómo se procesa el lenguaje en el cerebro humano, que es utilizado en el procesamiento de lenguaje natural dentro de la inteligencia artificial.

La neurociencia ha permitido obtener un conocimiento más profundo de la funcionalidad cerebral y de los procesos de procesamiento de información dentro del cerebro. Este conocimiento es valioso para los expertos en inteligencia artificial, debido a que les ofrece inspiración basada en los mecanismos y procesos cerebrales reales para desarrollar algoritmos y modelos en el campo del aprendizaje automático.

La neurociencia ha revelado principios fundamentales sobre cómo funcionan las redes neuronales en el cerebro y cómo este tiene la capacidad de cambiar y adaptarse a través del aprendizaje (plasticidad cerebral). Estos descubrimientos son importantes para el diseño de algoritmos en el aprendizaje automático, especialmente en el desarrollo de redes neuronales artificiales. Estos algoritmos buscan imitar los principios biológicos del cerebro humano para mejorar la eficiencia y capacidad de procesamiento en la inteligencia artificial.

La **ciencia cognitiva** es decisiva en el estudio de las actividades mentales humanas, como la percepción, el aprendizaje, la memoria, el pensamiento y la conciencia. Esta disciplina integra conocimientos de varias áreas de estudio como la psicología que contribuye con la comprensión del comportamiento y los procesos mentales; la lingüística que aporta conocimientos sobre el lenguaje y su relación con el pensamiento; la filosofía que ofrece marcos conceptuales y éticos; la neurociencia que brinda información sobre la estructura y función del cerebro; y la inteligencia artificial que ayuda a modelar y simular procesos cognitivos. Juntas, estas disciplinas permiten una comprensión más completa de cómo se desarrollan y trabajan las actividades mentales en los seres humanos.

La investigación en ciencia cognitiva busca descubrir cómo funcionan internamente los procesos a través de los cuales los seres humanos reciben, procesan y comprenden la información que obtienen de su entorno para desarrollar máquinas que puedan procesar e interpretar información sensorial (como imágenes, sonidos, texturas)

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

de manera similar a cómo lo hace un ser humano.

La investigación en ciencia cognitiva también ha explorado el pensamiento humano, incluyendo la resolución de problemas, la toma de decisiones y el razonamiento. Estos procesos cognitivos han sido estudiados y modelados para desarrollar algoritmos y sistemas de inteligencia artificial capaces de realizar tareas complejas de manera lógica y eficiente.

Los descubrimientos, que detallan los procesos mediante los cuales los humanos adquieren conocimientos (aprendizaje), los retienen (memoria) y los recuperan cuando es necesario, proporcionan una base sólida para la creación de modelos sofisticados en el campo del aprendizaje automático.

Se ha explorado el pensamiento humano en la búsqueda de entender cómo los humanos abordan y solucionan situaciones complejas, y cómo procesan información para llegar a conclusiones con el objetivo de crear máquinas con inteligencia artificial que imiten el comportamiento humano.

Comprender qué es la conciencia y cómo funciona sigue siendo un desafío en búsqueda de entender en profundidad la mente humana, a pesar de estos esfuerzos que se han hecho en la inteligencia artificial, simular completamente la conciencia humana es un desafío significativo y aún no se ha logrado. Los modelos actuales pueden aproximarse a ciertos aspectos de la conciencia, pero la experiencia subjetiva y la autoconciencia plena son elementos particularmente difíciles de replicar en una máquina (McDermott, 2007).

1.3 Areas de investigación de la inteligencia artificial

La inteligencia artificial (IA) representa un cambio significativo y un avance considerable en el campo de la tecnología informática, muestra el progreso que ha tenido en comparación con generaciones anteriores, que se centraban principalmente en realizar cálculos numéricos (como los usados en aplicaciones científicas o comerciales), actualmente la IA se orienta hacia tareas más complejas y abstractas, que incluyen la manipulación simbólica (trabajar con información representada mediante símbolos, como en el procesamiento del lenguaje) y la emulación de comportamientos inteligentes (intentar replicar o simular formas de inteligencia similares a las humanas, como el reconocimiento de patrones, la toma de decisiones, y el aprendizaje).

Las áreas de la IA en las cuales se está investigando en la actualidad son presentadas en la figura 1.2, se han agrupado partiendo de las formas de representación

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

del conocimiento en la inteligencia artificial que son: la representación simbólica y la representación no simbólica y también se puede observar combinadas de las dos representaciones.

La representación simbólica es un método para modelar y manipular el conocimiento utilizando símbolos (palabras, números o iconos) que se eligen y se utilizan de manera estructurada y lógica. Por ejemplo, en un sistema simbólico, una palabra como "elefante" = “mamífero paquidermo de gran tamaño” es un símbolo que representa un concepto específico en el mundo real.

La representación no simbólica, en lugar de usar reglas específicas y símbolos definidos, utiliza patrones y modelos matemáticos o estadísticos que aprenden de los ejemplos y datos. Por ejemplo, en una red neuronal utilizada para el reconocimiento de imágenes, no hay un símbolo específico para "elefante", sino que el conocimiento sobre qué constituyen estos animales se aprende (de muchas fotos de elefantes) y se almacena en los patrones de conexión entre las neuronas de la red.

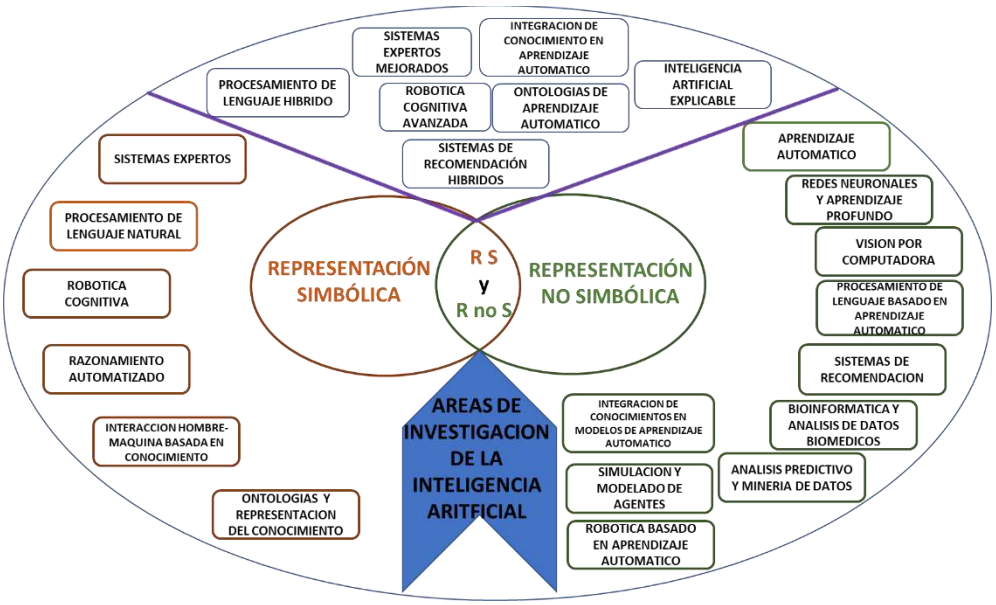


Figura 1.2 Áreas de investigación de la inteligencia artificial

Elaborado por los Autores

Se presenta una breve explicación de cada una de las áreas den investigación en inteligencia artificial:

Representación Simbólica:

1. Sistemas Expertos: Utilizan reglas para representar el conocimiento y la lógica para deducir nuevas informaciones.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

2. Procesamiento del Lenguaje Natural (PLN): Se enfoca en la comprensión y generación de lenguaje humano mediante reglas gramaticales y semánticas.
3. Robótica Cognitiva: Los robots son capaces de interpretar su entorno y reconocer los objetos y situaciones que encuentran, tienen la capacidad de interactuar físicamente con su entorno, tomando o moviendo los objetos (Franceschetti, 2018).
4. Razonamiento Automatizado: Las máquinas o sistemas informáticos son capaces de pensar y llegar a conclusiones de manera similar a como lo haría un ser humano, proceso que lo realizan de manera automática y estructurada mediante un programa o algoritmo.
5. Interacción Hombre-Máquina: Es el estudio y la parte práctica de cómo las personas se comunican o interactúan con las computadoras y otros dispositivos tecnológicos, buscando que sea fluido y sencillo como hablar o interactuar con otro ser humano.
6. Ontologías y Representación del Conocimiento: Se trata de crear modelos que puedan representar el conocimiento de manera que las computadoras lo entiendan y lo manejen eficientemente, utilizando entidades, atributos, y relaciones simbólicas (Poole & Mackworth, 2010).

Representación No-Simbólica:

1. Aprendizaje Automático: Se tienen los siguientes tipos de aprendizaje.

a) Aprendizaje Supervisado: Estos modelos de inteligencia artificial se utilizan para realizar tareas de clasificación o regresión, se entrenan usando un conjunto de datos que han sido etiquetados con anterioridad (Xiao, 2022).

b) Aprendizaje No Supervisado: Los modelos trabajan identificando patrones en datos que no han sido etiquetados, para locual usan técnicas de grupamiento o reducción de dimensionalidad.

c) Aprendizaje por Refuerzo: Son modelos (agentes) que aprenden a tomar decisiones mediante experimentación e interacción con el medio ambiente, reciben recompensas o penalizaciones en función de las acciones que realizan.

2. Redes Neuronales y Aprendizaje Profundo: Es un tipo de tecnología que intenta imitar la forma en que el cerebro humano procesa la información. Las redes neuronales son sometidas a un proceso llamado "entrenamiento", durante el cual

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

la red ajusta sus parámetros internos (llamados "pesos sinápticos") para mejorar su capacidad de realizar la tarea deseada (Haykin, 2009).

3. Visión por Computadora: Busca que las computadoras procesen y analicen el contenido visual (imágenes, videos) obtenido mediante cámaras y sensores del mundo que les rodea.
4. Procesamiento de Lenguaje Natural basado en Aprendizaje Automático: Usa modelos estadísticos y redes neuronales para el entendimiento, interpretación y generación de lenguaje humano de manera efectiva.
5. Sistemas de Recomendación: Son sistemas informáticos que se basan en algoritmos de aprendizaje profundo y estan diseñados para sugerir automáticamente productos, servicios, información o acciones a los usuarios.
6. Bioinformática y Análisis de Datos Biomédicos: Utiliza técnicas informáticas y estadísticas para entender y analizar los datos. La bioinformática se centra en el análisis de datos a nivel molecular, como los genes y proteínas, mientras que el análisis de datos biomédicos incluye datos relacionados con la salud y la biología.
7. Análisis Predictivo y Minería de Datos: Son técnicas que buscan patrones (como tendencias comunes o repetitivas), correlaciones (relaciones entre diferentes variables) y conocimientos a partir de grandes volúmenes de datos para predecir tendencias o comportamientos futuros.
8. Integración de Conocimiento en Modelos de Aprendizaje Automático: Se refiere al proceso de aprovechar la información y estructuras que ya existen he incorporárlas en los modelos de aprendizaje automático.
9. Simulación y Modelado Basado en Agentes: Se usa agentes autónomos para simular comportamientos y sistemas complejos y dinámicos. Cada agente en el modelo representa una entidad con su propio conjunto de características y reglas de comportamiento. Esta técnica es útil en el campo de la inteligencia artificial distribuida, donde se busca comprender y replicar comportamientos colectivos o de enjambre, como los observados en la naturaleza.
10. Robótica Basada en Aprendizaje Automático: Incorpora técnicas de aprendizaje automático en la robótica para mejorar las capacidades de percepción y decisión de los robots, especialmente en entornos que son dinámicos y no estructurados (Franceschetti, 2018).

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Representación Simbólica y Representación No-Simbólica:

1. Procesamiento del Lenguaje Natural (PLN) Híbrido: Estudia cómo hacer que las computadoras sean capaces de analizar, entender y generar lenguaje humano de forma automática. Usa métodos simbólicos como el análisis sintáctico basado en reglas gramaticales y la representación semántica basada en reglas. Combina con métodos no simbólicos, como el aprendizaje profundo. Se emplean en la traducción automática entre idiomas, el análisis de sentimientos y la generación de texto.
2. Sistemas Expertos Mejorados: Los sistemas expertos tradicionales, basados en conocimiento simbólico y reglas, se pueden enriquecer con técnicas de aprendizaje automático para manejar la incertidumbre, aprender patrones y capturen excepciones al manejar grandes volúmenes de datos.
3. Robótica Cognitiva Avanzada: Los robots utilizan algoritmos de aprendizaje profundo para interpretar lo que perciben a través de sus sensores y están equipados con sistemas que siguen reglas predefinidas para planificar sus acciones y tomar decisiones (Franceschetti, 2018). Estos robots pueden manejar una variedad de situaciones y tareas en diferentes entornos, realizando ajustes según sea necesario para adaptarse a cambios inesperados en su entorno.
4. Sistemas de Recomendación Híbridos: Combinan el análisis de patrones de comportamiento de usuarios (aprendizaje no simbólico) con reglas basadas en conocimiento específico del dominio (aprendizaje simbólico) para proporcionar recomendaciones más precisas y contextuales. Son asistentes inteligentes que te sugiere cosas basándose en lo que te gusta y en conocimientos específicos de un área determinada.
5. Integración de Conocimiento en Aprendizaje Automático: Se utiliza en proyectos que buscan integrar bases de conocimiento estructuradas (simbólicas) dentro de modelos de aprendizaje automático (no simbólicos) para mejorar la interpretación y generalización de los modelos.
6. Ontologías en Aprendizaje Automático: Se usa ontologías y estructuras de conocimiento simbólico para guiar y mejorar el proceso de aprendizaje en modelos no simbólicos, especialmente en dominios específicos como la medicina o la biología (García Serrano, 2012).

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

7. Inteligencia Artificial Explicable (XAI): Busca desarrollar modelos de IA, especialmente los basados en aprendizaje profundo, que puedan explicar sus decisiones y procesos de una manera comprensible para los humanos, a menudo integrando elementos simbólicos en la explicación de modelos no simbólicos.

1.4 Influencia significativa de la inteligencia artificial

La importancia de la inteligencia artificial radica en el hecho de que se relaciona con la mayoría de campos de investigación a través de diferentes disciplinas tales como: las ciencias de la ingeniería (ayuda a diseñar sistemas más eficientes y seguros), la economía (permite analizar tendencias de mercado y predecir cambios económicos) y la medicina (contribuye al diagnóstico y tratamiento de enfermedades, e incluso en la investigación de nuevos medicamentos), siendo usada la IA como un medio para hacer frente a muy complicados y difíciles procesos de cómputo, así como a los problemas cognitivos. Al ser la IA una de las principales corrientes de tratamiento de la información, puede ahora ofrecer soluciones a los problemas utilizando los avances e innovaciones entre una amplia gama de sub-áreas que inducen el pensamiento y razonamiento en los modelos y sistemas desarrollados.

Se espera que las empresas que utilizan aplicaciones de Inteligencia Artificial mejoren la capacidad de analizar los datos a través de múltiples variables, puedan detectar fraudes y gestionen de mejor manera las relaciones con los clientes para obtener una ventaja competitiva.

Altos directivos de muchas empresas utilizan sistemas de planificación estratégica basados en IA para tener una asistencia en funciones como: análisis de la competencia, el despliegue de tecnología y la asignación de recursos. También se utilizan programas para ayudar en el diseño de la configuración de equipos, distribución de productos, asesoramiento en el cumplimiento y evaluación de personal. La IA está contribuyendo en gran medida en la organización, planificación de la gestión y control de las operaciones, y continuará haciéndolo con más frecuencia a medida que se refinan los programas.

La IA también es influyente en la ciencia y la ingeniería, las aplicaciones desarrolladas se utilizan para organizar y manipular las cantidades cada vez mayores de información disponibles para los científicos e ingenieros, se emplea en clasificaciones biológicas, en la creación de circuitos semiconductores y componentes de automóviles. La IA se utiliza en la difracción y análisis de imágenes, en las plantas de energía y en el

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

diseño de las estaciones espaciales, uno de sus mayores usos es en la robótica.

Por la cantidad de habitantes que viven en el planeta tierra cada vez más se necesita usar de forma eficiente los recursos materiales y humanos, para lo cual podemos utilizar el poder que tienen las computadoras y la IA, en la agricultura para controlar plagas y manejar cultivos en forma más eficiente; en las fábricas en la realización de montajes peligrosos y actividades tediosas (labores de inspección y mantenimiento); en medicina en el diagnóstico a pacientes, detectar pacientes que están en mayor riesgo de complicaciones, supervisar la condición de los pacientes, descubrir sutiles interacciones entre los medicamentos que ponen a los pacientes en riesgo de sufrir efectos secundarios graves, administrar tratamientos y preparar estudios estadísticos; en el trabajo doméstico asistir en la limpieza de la casa, preparación de alimentos, a brindar asesoría acerca de dietas, compras, supervisión y gestión de consumo energético y seguridad del hogar; en las escuelas apoyar en la formación de los estudiantes, especialmente en aquellas materias consideradas complejas; colaborar con los humanos expertos en el análisis para la solución de problemas difíciles o en el diseño de nuevos dispositivos.

Se ha convertido en algo común de nuestras vidas el uso del GPS (Global Positioning System) para escoger la mejor ruta, recibir estados del tráfico, usar teléfonos inteligentes que entienden nuestro lenguaje, asistentes inteligentes como Cortana de Microsoft y Siri de Apple; la búsqueda inteligente que realiza el buscador de Google y Bing, el traductor y lector de documentos de Google. Algoritmos inteligentes que detectan los rostros mientras estamos tomando una foto con el celular y reconocen las caras de las personas cuando se publican las fotos en Facebook. Los usuarios de Amazon reciben recomendaciones sobre libros, los de Netflix sobre películas. Aceleración de las portátiles al adivinar lo que se hará a continuación. Los coches inteligentes que se conducen a sí mismos, que pueden enfrentar situaciones complicadas, que reconocen la voz para una interacción natural, que están desarrollando BMW, Tesla, Google. La lucha contra el spam (correo basura) a través de clasificadores bayesianos y otras técnicas probabilísticas han mostrado su eficacia. Los militares están usando aviones no tripulados, drones y robots para investigar en lugares peligrosos.

En la actualidad existe dispositivos con IA que permiten que la gente ciega vea, los sordos escuchen, y los discapacitados y ancianos puedan caminar, e incluso correr. Las galaxias están siendo exploradas en mutuo apoyo entre los astrónomos

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

experimentados y las máquinas.

La presencia de los métodos de Inteligencia artificial ha puesto a los seres humanos a pensar en los riesgos potenciales de sus avances, piensan algunas personas que los sistemas de IA se podrían convertir en superinteligentes y poner en peligro la supervivencia de la humanidad, pero se tiene a la AAAI (Association for the Advancement of Artificial Intelligence) que está encargada de incentivar el avance en la ciencia y tecnología de la inteligencia artificial y promocionar su uso responsable.

Los errores de programación en software de inteligencia artificial deben ser considerados, por lo que se requiere de la verificación y validación del software debido a la creciente complejidad y el uso en funciones delicadas como en automóviles, robots quirúrgicos y sistemas de armas. Un desafío técnico es garantizar que los sistemas integrados de forma automática a través de métodos estadísticos se comporten correctamente en el proceso de aprendizaje automático. Otro desafío es garantizar el buen comportamiento cuando un sistema de IA se encuentra con situaciones imprevistas. Los vehículos automatizados, robots para el hogar, y los servicios inteligentes en la nube deben funcionar bien, incluso cuando reciben insumos sorprendentes o confusos (Lalanda et al., 2013).

Otro riesgo que está presente son los ciberataques a los cuales los programas de IA son también vulnerables, y si se piensa en lo que podría suceder al estar los algoritmos de IA encargados de tomar decisiones de alto poder. Por lo que se requiere la investigación en seguridad cibernética para estar seguros que los algoritmos de IA y los sistemas que se desarrollan basados en estos pueden sobrevivir los ataques cibernéticos a gran escala.

Un aspecto importante de cualquier sistema de IA que interactúa con la gente es que debe razonar sobre lo que las personas piensan en lugar de llevar a cabo los comandos de manera literal, para juzgar si lo que se le pide que realice es normal o razonable para la mayoría de la gente.

1.5 Turing y la Evaluación de la Inteligencia de las Máquinas

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Figura 1.3 Alan Turing

Fuente: Elliott & Fry

Turing (1912-1954) propuso la Máquina Niño, en la que se crearía un agente inteligente básico y se sometería a un curso de educación para proporcionarle conocimiento. En su artículo de 1950, "Computing machinery and intelligence", Turing (Figura 1.3) argumentó que si una máquina puede actuar como un humano, entonces se puede considerar que es inteligente (Turing, 2009). El Test de Turing se diseñó para proporcionar una definición operacional y satisfactoria de inteligencia, basada en la incapacidad de diferenciar entre entidades inteligentes indiscutibles y seres humanos (Floridi, 2023). Según este test, un computador supera la prueba si un evaluador humano no puede distinguir si las respuestas a una serie de preguntas planteadas son de una persona o no.

La prueba de Turing tiene mucha importancia en la investigación y el desarrollo de la inteligencia artificial, debido a que permite realizar la evaluación de la capacidad de una máquina de imitar el comportamiento humano. A continuación se indica los requerimientos de la prueba de Turing:

El ***Procesamiento de Lenguaje Natural*** (NLP Natural Language Processing) capacita a las computadoras para comunicarse eficazmente en inglés y otros idiomas. Las aplicaciones del NLP van mas allá de la traducción automática, extendiéndose al reconocimiento y comprensión del lenguaje humano, en búsqueda de que las máquinas pueden mantener conversaciones que sean indistinguibles de las que mantienen los humanos.

La ***Representación del Conocimiento*** permite poder almacenar lo que se conoce o siente, la computadora debe ser capaz de almacenar y recuperar de forma eficiente la información que va obteniendo e infiriendo autónomamente. En la actualidad se investigan las técnicas de almacenamiento de información para que exista fácil accesibilidad y sea utilizable por los sistemas inteligentes.

El ***Razonamiento automático*** utiliza la información almacenada para responder a preguntas y extraer nuevas conclusiones, esta temática se trata a través de los sistemas expertos que buscan llegar a conclusiones lógicas a partir de hechos o premisas introducidas inicialmente en el sistema; otra forma es a través de las redes probabilísticas como las redes Bayesianas o de Markov que permiten realizar predicciones y llegar a

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

conclusiones aun cuando existe incertidumbre.

El ***Aprendizaje automático*** permite a las máquinas adaptarse a nuevas circunstancias, detectar y extrapolar patrones. Permite que las máquinas sean capaces de aprender cosas nuevas, de adaptarse al medio; para lo cual en la actualidad se usan técnicas basadas en redes neuronales y métodos probabilísticos como las redes bayesianas o de Markov (Croitoru et al., 2018).

La ***Visión computacional***, es una disciplina esencial, porque capacita a sistemas inteligentes para percibir objetos y comprender su entorno a través de imágenes. Su desarrollo persigue la ambiciosa meta de igualar la capacidad de percepción visual humana utilizando medios electrónicos para interpretar y comprender imágenes. La comprensión de la imagen se puede considerar como el proceso de esclareser información simbólica a partir de los datos visuales, empleando modelos que se fundamentan en principios de geometría, física, estadística y teoría del aprendizaje.

La ***Robótica***, usada para manipular y mover objetos, tomando en cuenta el peso, la forma, la presión que debe aplicarse para no dañar el objeto y saber los movimientos a realizar para trasladar el objeto a su destino. En el presente constituyen máquinas automatizadas que pueden tomar el lugar de los humanos en entornos peligrosos o procesos de fabricación, o parecerse a los humanos en apariencia en el comportamiento y la cognición.

1.6 Agentes inteligentes

El paradigma principal en el campo de la Inteligencia Artificial es el modelo racional, que sostiene que una máquina es considerada inteligente si piensa y se comporta de manera racional. Este enfoque se apoya en técnicas de lógica y en el concepto de agentes desarrollado por Russell y Norvig. Los agentes toman decisiones basadas en el razonamiento y las conclusiones extraídas de la información disponible, priorizando la opción más conveniente en función de los datos y el tiempo disponible.

El agente humano tiene un cuerpo con un conjunto de sensores que incluye la visión, la audición, el olfato, el gusto, la piel, el sentido vestibular para el equilibrio, la nocicepción relacionada al dolor, también tiene una serie de sistemas actuadores que incluyen dedos, brazos y piernas y otros sistemas de control motores como el cerebro y el sistema nervioso central.

En el ámbito de la Inteligencia Artificial, un agente inteligente, también conocido

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

como *agente*, es una entidad autónoma que existe en un entorno y se guía por la información que recibe a través de sensores. Actúa de manera racional utilizando actuadores para llevar a cabo acciones. En un instante dado, un agente toma decisiones basadas en la secuencia completa de percepciones que ha recibido hasta ese momento. Para realizar esta tarea, utiliza un sistema de control que mapea las percepciones en acciones (figura 1.4). Un ejemplo de agente sería un automóvil autónomo que debe tomar decisiones para llegar a su destino.

En el ámbito de la Inteligencia Artificial, el sistema de control de un agente se refiere al algoritmo avanzado responsable de convertir percepciones captadas a través de sensores en acciones específicas. Este algoritmo opera dentro de una arquitectura análoga a la de una computadora, dotada de sensores físicos y actuadores para interactuar con el entorno. Un Agente Inteligente se define como:

Agente = Arquitectura + Programa

El sistema de control es esencial para el funcionamiento del agente, debido a que permite que este interprete la información del entorno a través de los sensores y tome decisiones lógicas sobre las acciones a emprender utilizando los actuadores. Es como el cerebro del agente, procesando la información entrante y generando respuestas adecuadas en base a su programación.

La arquitectura computacional permite que el agente interactúe con el medio ambiente de manera autónoma y realice las tareas para las que fue diseñado. Los sensores permiten capturar datos del entorno y se transmiten al sistema de control, que analiza y toma decisiones en función de la programación establecida, para proceder a continuación, a envíar las señales apropiadas a los actuadores, que ejecutan las acciones correspondientes en el medio ambiente.

La arquitectura más simple puede ser la *computadora* que posee un teclado y un mouse que constituyen los sensores y tiene el actuador que es el monitor. Las percepciones de los sensores son utilizadas por el programa quien se encarga que las acciones generadas se muestren a través de la pantalla del monitor (Leija, 2021). Para escribir el programa se requiere un lenguaje de programación como lisp, prolog, phyton, C++, java, matlab, etc.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

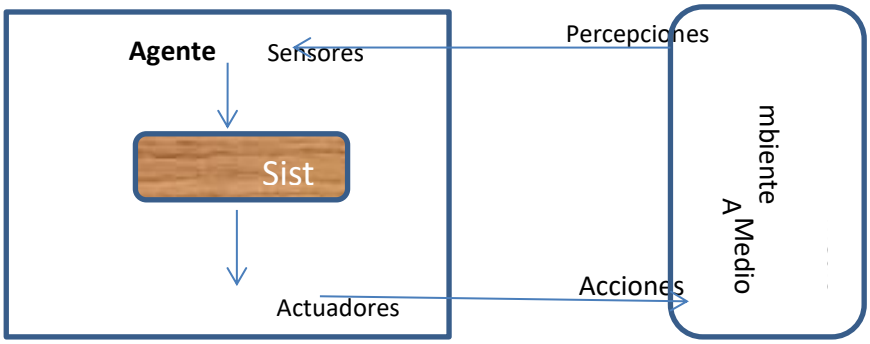


Figura 1.4 Esquema de un agente

Elaborado por los Autores

La Inteligencia Artificial se enfoca en el desarrollo de programas que puedan mostrar un comportamiento racional, utilizando una pequeña cantidad de código en lugar de tener una tabla con un gran número de entradas. Un ejemplo de esto es el cálculo de las raíces cuadradas, en el pasado, las personas utilizaban tablas con los resultados de las raíces cuadradas para diferentes números. Sin embargo, con el avance de la IA, se ha logrado reemplazar estas tablas por programas que tienen unas pocas líneas en las calculadoras electrónicas, lo que permite calcular fácilmente la raíz cuadrada de cualquier número.

Este enfoque basado en programas más compactos y generales es un objetivo de la IA. En lugar de tener que almacenar y consultar una gran cantidad de datos específicos, se busca desarrollar algoritmos que puedan inferir y generalizar patrones a partir de información limitada. Esto permite que los programas de IA sean más flexibles y capaces de abordar una amplia gama de situaciones sin requerir una lista exhaustiva de casos específicos.

1.7 Diseño del agente

El agente racional se esfuerza por tomar decisiones acertadas en el momento oportuno, con el objetivo de lograr resultados óptimos. Para evaluar el desempeño de un agente en su entorno, es fundamental medir su éxito en función de la secuencia de acciones realizadas y las percepciones recibidas. Un estado se refiere a una configuración particular del agente en su entorno, mientras que el estado inicial indica el punto de partida del agente.

Las acciones que toma el agente conducen a una secuencia de estados a medida que interactúa con su medio ambiente, si la secuencia coincide con la secuencia deseada, se considera que el agente ha actuado correctamente. Las acciones son simplemente

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

opciones que pueden elegirse en cualquier estado dado, y su elección tiene un impacto directo en cómo evoluciona el agente en el entorno.

Un agente racional debe tomar decisiones informadas y eficientes, considerando las percepciones y el conocimiento disponible, con el fin de maximizar la consecución de sus objetivos. Al observar el entorno y tomar de decisiones adecuadas, el agente puede adaptarse y aprender a actuar de manera más inteligente en situaciones futuras.

La capacidad de un agente para seleccionar las acciones más apropiadas en cada estado y lograr un rendimiento satisfactorio es un desafío importante en el campo de la inteligencia artificial.

Los investigadores y expertos en IA se dedican a desarrollar algoritmos y técnicas que permitan a los agentes racionales tomar decisiones óptimas en una amplia variedad de situaciones, contribuyendo así al avance y aplicación de esta disciplina.

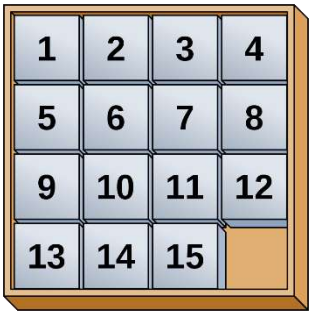


Figura 1.5 Rompecabezas

Fuente: Wikipedia

Un estado se refiere a una configuración específica en la que se encuentra un agente dentro de su entorno. Tomemos como ejemplo un rompecabezas (figura 1.5) de 15 piezas: cada estado representa una disposición particular en la que los números están ordenados en el tablero. Las acciones, por otro lado, son las elecciones que se pueden realizar en un estado dado.

Cuando se recibe un estado como entrada, el sistema devuelve como salida el conjunto de acciones que se pueden llevar a cabo en ese estado en particular. Siguiendo el ejemplo del rompecabezas de 15, las acciones disponibles en un estado dado están determinadas por las formas en que los cuadrados se pueden deslizar dentro de la configuración actual. Por ejemplo, si el cuadrado vacío se encuentra en el medio, hay 4 acciones posibles; si está junto a un borde, hay 3 acciones disponibles; y si está en una esquina, solo hay 2 acciones posibles.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Estas acciones permiten al agente interactuar y modificar su estado en el entorno, buscando lograr un objetivo específico. La capacidad de seleccionar las acciones más adecuadas en función de un estado dado es esencial para que el agente resuelva problemas, tome decisiones informadas y maximice su rendimiento en el entorno en el que se encuentra.

En el campo de la Inteligencia Artificial, se desarrollan algoritmos y técnicas que permiten a los agentes racionales determinar de manera eficiente las acciones disponibles en cada estado y tomar decisiones óptimas para alcanzar sus metas. El estudio y avance de estas capacidades de toma de decisiones en la IA contribuyen al desarrollo de sistemas más inteligentes y eficientes en diversos dominios de aplicación.

La secuencia de estados se muestra a continuación con otro ejemplo. Se tienen tres recipientes (figura 1.6), cuyas capacidades son doce litros, ocho litros y tres litros, además un grifo de agua. Se desea obtener un litro de agua utilizando los recipientes, para lo cual puede llenar los recipientes o vaciar de un recipiente a otro, o vaciar el agua de un recipiente en el suelo. Existen algunas secuencias de estados que se pueden seguir, a continuación, se muestra una de ellas.

Figura 1.6 Tamaño de recipientes en litros

Elaborado por los Autores

[12,8,3] Tamaño de los tres recipientes en litros

[0,0,0] Todos los recipientes en un inicio se encuentran vacíos (Estado inicial)

[12,0,0] El recipiente de 12 litros se llena con agua del grifo (Estado intermedio)

[4,8,0] Se vierte el agua del recipiente de 12 litros en el de 8 litros (Estado intermedio)

[1,8,3] Se vierte el agua del recipiente de 12 litros que contiene 4 litros en el de 3litros

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

(Estado intermedio)

[1,0,0] En el recipiente de 12 litros queda el 1 litro de agua, como las otras cantidades de líquido no son las buscadas se derrama en el suelo (Estado final)

Este ejemplo muestra una secuencia de estados que nos permite obtener un litro de agua utilizando los recipientes mencionados. Cada estado representa una configuración específica de los recipientes y las acciones realizadas (llenar, transferir, vaciar) nos permiten llegar al estado deseado. Este tipo de problemas de razonamiento y planificación son abordados mediante técnicas de inteligencia artificial, que permiten encontrar soluciones óptimas o aproximadas a través de algoritmos específicos.

La capacidad de representar y manipular estados, así como de planificar secuencias de acciones para lograr un objetivo, es fundamental en el campo de la IA. Estas habilidades permiten a los sistemas inteligentes resolver problemas complejos y tomar decisiones informadas en una amplia variedad de dominios, desde problemas de optimización hasta tareas de planificación y control.

Para el diseño del agente se establecen cuatro factores que se describen a continuación en la tabla 1-1.

Tabla 1-1 Factores de diseño

Factores de diseño	**Conceptualización**
Medida de rendimiento	Quien define la medida de rendimiento es el diseñador del agente, el cual establece los requisitos y objetivos específicos en función del entorno donde va a trabajar el agente, incluyendo métricas como la precisión (qué tan correctas son las acciones o respuestas del agente), eficiencia (qué cantidad de recursos que consume el agente para realizar su tarea), velocidad de reacción(qué tan rápido puede responder o actuar el agente) y la satisfacción del usuario (cómo se sienten los usuarios acerca de las interacciones con el agente). Estas métricas sirven para que el diseñador ajuste y calibre adecuadamente el agente.
Entorno de trabajo	El entorno de trabajo es la estructura física, lógica y digital donde el agente realiza sus tareas y toma decisiones, es el lugar donde interactúa con objetos y entidades presentes (otros agentes, humanos,

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	herramientas); el diseñador debe identificar posibles problemas que el agente tiene que enfrentar y proceder a desarrollar estrategias y algoritmos que permitan que pueda el agente realizar sus tareas de manera eficiente y competitiva.
Percepciones	Las percepciones es la capacidad del agente para obtener información del medio ambiente que lo rodea a través de sensores. Los sensores son los dispositivos que permiten al agente captar datos y señales del entorno, que le proporcionan información al agente sobre diferentes aspectos como el estado físico, la presencia de objetos, eventos o cualquier otro tipo de estímulo relevante. Mediante estos datos el agente pueda comprender y tomar decisiones adecuadas en su entorno.
Acciones	Las acciones representan el efecto que tiene el agente sobre su medio ambiente, utilizando actuadores, estos son dispositivos o mecanismos que permiten al agente interactuar físicamente con el entorno y realizar cambios o ejecutar acciones específicas. Estos actuadores pueden variar según el tipo de agente y la aplicación para la cual están destinados, e incluir elementos como motores, brazos robóticos, sistemas de manipulación, entre otros. La capacidad de realizar acciones al agente le permite manifestar su comportamiento y llevar a cabo tareas en su entorno. El agente mediante los actuadores, puede realizar acciones como mover objetos, desplazarse en el espacio, modificar configuraciones o ejecutar cualquier otra acción física requerida en el contexto.

Elaborado por los Autores

En el diseño del agente denominado robot que juega al futbol (figura 1.7) la medida de rendimiento constituye: partidos ganados, partidos empatados, partidos perdidos, goles a favor, goles en contra. El entorno de trabajo va a ser el campo de juego, la pelota, sus compañeros de juego, los robots adversarios. Para que pueda realizar las percepciones se requiere una cámara, sensores táctiles, acelerómetro, sensores de orientación. Para efectuar acciones necesita dispositivos de locomoción como las piernas que le permitan trasladarse de un lugar a otro y dispositivos para patear el balón.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Figura 1.7 Robots jugando futbol

Fuente: El País España

El agente racional además de recopilar información debe aprender lo máximo posible de lo que está percibiendo, la configuración inicial del agente debe reflejar un conocimiento preliminar del entorno, pero a medida que adquiere experiencia el conocimiento se debe modificar y aumentar.

Las funciones del agente se calculan en tres períodos que son:

- Durante el diseño de agente: Los encargados de diseñar el agente realizan cálculos, para que el agente posea el conocimiento inicial requerido.
- El agente pensando en el próximo estado: Cuando el agente piensa en la siguiente operación, realiza más cálculos.
- El agente aprende de la experiencia: Producto de la experiencia el agente lleva a cabo más cálculos para decidir cómo modificar su forma de comportarse.

El agente racional debe tener un comportamiento autónomo que le permita aprender a determinar cómo compensar el conocimiento incompleto o parcial inicial, pudiendo de esta forma ser capaz de navegar en su entorno sin la guía de una entidad externa (un operador humano), pueda buscar metas en su entorno, resolver problemas.

El agente debe poder subsistir a través del tiempo en el medio ambiente que se lo ubique, e interactuar con otros agentes para proporcionar información o que le proporcionen información, o comunicarse con los usuarios. Tener movilidad le permite al agente migrar entre sistemas a través de una red, tener la capacidad de aprender y adaptarse al entorno. En el proceso de aprendizaje los sensores deben mapear el entorno, para que el sistema de control de las órdenes adecuadas a los actuadores quienes demuestran una conducta o comportamiento inteligente que

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

satisfaga un conjunto de restricciones definidas.

Después de las suficientes experiencias interaccionando con el entorno, el comportamiento del agente racional será independiente del conocimiento que poseía inicialmente. Esto significa que el agente ha adquirido la capacidad de adaptarse y responder de manera inteligente a diferentes situaciones sin depender únicamente de su conocimiento previo.

1.7.1 Propiedades del entorno de trabajo

Las propiedades del entorno de trabajo se refieren a las características y cualidades del ambiente en donde un agente o sistema de inteligencia artificial ejecuta sus acciones. Las propiedades del entorno influyen en cómo un agente puede percibir y actuar, y pueden variar ampliamente según la aplicación específica. Es necesario categorizar los entornos de trabajo (Tabla 1-2) en los que el agente inteligente va a desenvolverse, esto ayuda en su diseño.

Tabla 1-2 Propiedades del entorno de trabajo y ejemplos de agentes

ENTORNO	CARACTERÍSTICAS	EJEMPLOS DE AGENTES
Totalmente observable	En un entorno totalmente observable, los sensores del agente deben proporcionar una representación completa y fiable del estado del medio ambiente en que se desenvuelve, en cada momento. El agente debe detectar y capturar todos los aspectos que son relevantes en la toma de decisiones que están en función de las medidas de rendimiento (MacKay, 2003). En estos entornos el agente no necesita mantener ningún estado interno para saber qué sucede en el mundo. Al existir una visión integral y detallada del estado actual, se facilita la toma de decisiones informadas sin	○ Ayudante de matemáticas ○ Juego de Ajedrez ○ Sistema de Navegación GPS ○ Laberinto con visibilidad completa ○ Sistema de Monitoreo de Sensores ○ Agente de reconocimiento de voz ○ seguimiento de objetos ○ análisis de datos

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	incertidumbre, el agente no requiere mantener un histórico para completar la información faltante o inferir aspectos ocultos del entorno. La capacidad de observación completa y sin limitaciones brinda al agente una ventaja al comprender de manera precisa el contexto en el que actúa, poder tomar decisiones adecuadas en tiempo real y realizar cálculos exactos del beneficio, importancia de una determinada acción, o la recompensa esperada basado en el estado observable.	○ control de calidad ○ reconocimiento facial
Parcialmente observable	En los entornos parcialmente observables la percepción del agente es afectada por la presencia de ruido (Lindley, 2013) o la utilización de sensores poco exactos o porque el agente no recibe información completa del sistema; esta percepción limitada o incompleta le dificulta la obtención de una visión global y precisa del estado del entorno, pudiendo generar incertidumbre en el agente. El agente al no tener acceso a una representación directa y completa del estado en el que se encuentra, se ve obligado a realizar inferencias y estimaciones para obtener una comprensión aproximada del estado actual. En búsqueda de enfrentar esta	○ Poker ○ Tráfico en tiempo real ○ Laberinto con visibilidad limitada ○ Predicción del clima ○ Robot que juega al futbol ○ Juego del veintiuno (Black jack). ○ Robot clasificador de baldosas ○ Robot de limpieza autónomo ○ Asistente virtual inteligente ○ reconocimiento de objetos

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	situación se debe incluir la utilización de algoritmos de filtrado que procesen secuencias de datos ruidosos o incompletos y técnicas de fusión de datos que combinen la información entregada por los sensores de tal forma que se pueda mejorar la precisión de la percepción.	○ Sistema de detección y prevención de fraudes ○ diagnóstico médico
Determinista	En los entornos deterministas, el siguiente estado del medio ambiente está totalmente determinado por la acción realizada en el estado actual. Al ejecutar una acción específica, el resultado o estado siguiente del entorno es predecible de manera precisa y no hay incertidumbre involucrada (Samuel, 1959). La relación causa-efecto entre las acciones del agente y los estados del entorno, nos indica, que cualquier acción que el agente realice tiene un resultado específico y directo en el entorno.	○ Ayudante de matemáticas ○ Juego de ajedrez ○ Laberinto sin obstáculos móviles ○ Operaciones matemáticas ○ Control de sistemas físicos
Estocástico	En los entornos estocásticos, el cambio del entorno está sujeto a cierto grado de aleatoriedad o incertidumbre debido a las acciones del agente; aunque el agente tome una acción específica, el resultado o estado siguiente del entorno no pueda predecirse con total certeza, este puede variar debido a factores aleatorios o no completamente conocidos así se	○ Robot que juega al futbol. ○ Robot clasificador de baldosas ○ Mercados financieros ○ Tráfico de vehículos ○ Clima ○ Juego de póker

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	conozca el estado actual. Debido a la incertidumbre presente en el entorno estocástico, la probabilidad de los diferentes resultados puede modelarse utilizando modelos probabilísticos para tomar decisiones informadas y racionales.	
Estratégico	En los entornos estratégicos, el medio ambiente es determinista para el agente, lo que significa que el estado siguiente del entorno está completamente determinado por las acciones del agente. Al existir otros agentes en el entorno se introduce cierto nivel de incertidumbre estratégica, debido a que las acciones de los otros agentes pueden influir en el estado siguiente del entorno de manera impredecible; esto requiere que el agente considere diferentes estrategias y tome decisiones adaptativas para enfrentar los posibles resultados de las acciones de los otros agentes. Para lograr el agente sus objetivos en estos entornos que son dinámicos y competitivos, deben poder analizar y anticipar las acciones de los otros agentes.	○ Juego del veintiuno (black jack) ○ Ajedrez ○ Poker ○ Negociación y subastas ○ Competencia empresarial ○ Estrategia militar ○ Videojuegos de estrategia en tiempo real (RTS) ○ Planificación logística ○ Sistemas de gestión de tráfico ○ Simuladores de vuelo
Episódico	En un entorno episódico, cada estado en el que se encuentra el agente no depende de los estados anteriores, por	○ Búsqueda en laberintos ○ Robot clasificador

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	lo que una acción realizada en un estado no tiene influencia en los estados futuros, cada estado constituye un episodio atómico. El agente no tiene una memoria de larga duración y cada interacción con el entorno se considera como un episodio separado e independiente. Para la toma de decisiones y realizar acciones el agente se basa únicamente en la información disponible en el estado actual, sin tener en cuenta el historial de estados o acciones pasadas. Esto simplifica el modelado y la solución de problemas en entornos en los que la secuencia de acciones no tiene una influencia acumulativa en los resultados futuros.	de baldosas ○ Recolección de objetos ○ Juegos de cartas como el Solitario ○ Juegos de plataformas (Super Mario Bros) ○ Juegos de escape ○ Simulaciones de emergencias ○ Juegos de rol (Dungeons & Dragons) ○ Juegos de lógica y acertijos
Secuencial	En los entornos secuenciales, las decisiones tomadas en un momento dado pueden afectar a las decisiones y resultados futuros, cada estado depende del histórico anterior. El agente debe evaluar no solo el impacto inmediato que tienen sus acciones, sino también considerar cuidadosamente las consecuencias de sus acciones y tomar decisiones estratégicas para maximizar su rendimiento total en el transcurso del tiempo. La secuencialidad y la necesidad de	○ Navegación en un entorno desconocido ○ Juegos de ajedrez ○ Toma de decisiones en tiempo real ○ Robot que juega al futbol ○ Ayudante de matemáticas ○ Juego del veintiuno (Black jack) ○ Sistemas de control de inventario

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	planificación a largo plazo hacen que estos entornos de trabajo sean más desafiantes y requieran algoritmos y enfoques especializados para la toma de decisiones y el control inteligente del agente, que maximice los resultados positivos a lo largo del tiempo.	○ Planificación y programación de rutas ○ Juegos de aventuras o exploración
Estático	En un entorno estático, mientras el agente delibera sobre la acción siguiente el entorno no cambia. El agente no tiene que enfrentarse con cambios inesperados o dinámicos en su entorno mientras toma decisiones o realiza acciones. Los entornos estáticos son más fáciles de entender y predecir, facilitando el diseño de los agentes y algoritmos que funcionan eficientemente en estos entornos.	○ Juego del veintiuno (Black jack) ○ Rompecabezas ○ Laberinto ○ Clasificación de imágenes ○ Sistemas de detección de spam ○ Sistemas de filtrado de contenido ○ Sistemas de diagnóstico médico ○ Sistemas de recomendación de productos ○ Análisis de datos financieros
Dinámico	En los entornos dinámicos, el entorno puede experimentar cambios mientras el agente está considerando la acción futura que va a realizar. Estos cambios pueden ser el resultado de eventos externos, interacciones con otros agentes o incluso como resultado	○ Robot que juega al futbol ○ Robot clasificador de baldosas ○ Tráfico de vehículos ○ Juego de ajedrez

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	sus propias acciones. El agente, debe poder adaptarse y tomar decisiones en tiempo real a medida que el entorno evoluciona. Debe ser capaz de detectar y comprender los cambios en su entorno, actualizar su conocimiento y ajustar su comportamiento en consecuencia. Para hacer frente a los entornos dinámicos, los agentes utilizan técnicas como la monitorización constante del entorno, la detección de cambios, la planificación en tiempo real y el aprendizaje adaptativo. Estas técnicas hacen que el agente tome decisiones informadas y adapte su estrategia a los cambios del entorno.	○ Simulaciones de vuelo ○ Sistemas de atención médica personalizada ○ Sistemas de recomendación personalizada ○ Sistemas de simulación de vida artificial ○ Sistemas de gestión de energía ○ Sistemas de logística y distribución
Semiestático	El entorno semiestático no experimenta cambios significativos, se mantiene relativamente constante a lo largo del tiempo en términos de sus características físicas y de configuración. Sin embargo, el rendimiento puede variar a medida que se desarrolla el agente, esto se debe a la capacidad que tiene de aprender, adaptarse y mejorar en el tiempo. El agente puede también experimentar mejoras en su capacidad de tomar decisiones, resolver problemas o alcanzar sus objetivos a	○ Ayudante de matemáticas ○ Juegos de mesa ○ Sistemas de gestión de inventario ○ Planificación de rutas de transporte ○ Configuración de redes ○ Sistemas de recomendación de contenido ○ Sistemas de gestión de recursos

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	medida que adquiere experiencia y conocimiento. Los agentes para mejorar su rendimiento utilizan técnicas de aprendizaje automático, optimización y toma de decisiones basadas en datos.	naturales o Sistemas de planificación urbana o Sistemas de optimización de energía o Sistemas de atención al cliente
Discreto	En los entornos discretos, el conjunto de estados, las acciones y percepciones que puede experimentar el agente, como el espacio de búsqueda, son finitos y claramente definidos. El agente tiene un número limitado y específico de estados en los puede encontrarse, así como tambien un conjunto definido de acciones y percepciones que puede realizar y recibir. La discreción en los entornos proporciona una estructura clara y bien definida, que facilita la modelización y el análisis del comportamiento del agente. Los estados, acciones y percepciones se pueden representar mediante variables discretas o categorías específicas, que simplifica el diseño y la implementación de algoritmos y estrategias de toma de decisiones. Al tener un número finito de estados, acciones y percepciones, es posible	o Ayudante de matemáticas o Juego del veintiuno (Black jack) o Juego de ajedrez o Sistema de navegación en una cuadrícula o Solución de puzles o Juego de laberintos o Juego de estrategia por turnos o Juego de palabras o Tres en raya o Juego de damas o Solución de crucigramas o Simulación de vida artificial o Simulación de tráfico peatonal o Control de inventario o Sistemas de control

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	realizar un mapeo detallado y sistemático de todas las posibles combinaciones, lo que permite una planificación precisa y una evaluación completa de las estrategias del agente. La naturaleza discreta de los entornos proporciona una base sólida para la aplicación de técnicas matemáticas, algoritmos de búsqueda, optimización y aprendizaje. Esto incluye métodos como los árboles de búsqueda, algoritmos de programación dinámica y técnicas de aprendizaje automático basadas en modelos discretos.	de producción
Continuo	En los entornos continuos, el número de estados posibles es extremadamente amplio e incluso infinito. En el transcurso del tiempo, los estados del medio ambiente experimentan cambios continuos, esto hace que la percepción del agente en cada instante sea única y varíe de forma continua. En consecuencia, el agente se ve obligado tomar decisiones y acciones continuas para adaptarse y responder a estos cambios dinámicos. Dado que el número de estados es extenso, resulta imposible mapear y explorar exhaustivamente todas las posibilidades. Para poder abordar problemas en entornos continuos es necesario utilizar enfoques basados en	○ Robot que juega al futbol. ○ Robot clasificador de baldosas. ○ Sistema de control de vuelo de un avión ○ Brazo robótico ○ Sistema de navegación autónoma de vehículos ○ Simulaciones de física ○ Control de procesos industriales ○ Gestión de carteras de inversión ○ Control de sistemas

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	técnicas como el análisis de funciones de valor, algoritmos de optimización numérica y métodos basados en modelos probabilísticos; que permiten representar y aproximar los estados y acciones de manera continua, esto facilita la toma de decisiones mediante la maximización de una función objetivo o la adaptación a patrones y cambios en el medio ambiente. En los entornos continuos, el aprendizaje y la adaptación es importante. Los agentes inteligentes deben ser capaces de aprender de la experiencia y mejorar sus acciones en función de la retroalimentación continúa proporcionada por el entorno (Gelfond & Kahl, 2014). Es necesario el uso de técnicas de aprendizaje automático y algoritmos de control adaptativo para ajustar y mejorar su rendimiento a medida que interactúan con el medio ambiente.	de energía renovable o Robot de manipulación de objetos o Sistema de control de tráfico aéreo o Simulaciones de fluidos o Optimización de carreteras y transporte o Sistemas de recomendación personalizados o Control de sistemas de energía renovable
Agente individual	En los entornos de agente individual, solo un agente interactúa de manera autónoma con el medio ambiente. Por lo tanto, no hay necesidad de interacción o colaboración directa con otros agentes, centrándose toda la actividad en el agente principal. En estos escenarios, el agente individual asume la responsabilidad	o Asistente virtual en un teléfono inteligente o Robot de limpieza o Sistema de recomendación de películas o Agente de comercio electrónico

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	completa de percibir su entorno, tomar decisiones y ejecutar acciones que se alinien con su objetivo y las condiciones específicas del entorno. La ausencia de interacciones directas con otros agentes reduce la complejidad de las dinámicas de interacción y las estrategias a considerar. Aunque el agente trabaja de forma individual, puede estar inmerso en esenarios donde existen otros agentes de forma indirecta, como sistemas automatizados o programas de software que son importantes en la configuración o en el control del entorno. Aunque no interactúan directamente con el agente principal, estos elementos pueden afectar significativamente las condiciones y restricciones del entorno, perjudicando las decisiones y acciones del agente individual.	○ Chatbot de servicio al cliente ○ Robot clasificador de baldosas ○ Asistente personal en un dispositivo doméstico inteligente ○ Agente de salud personal ○ Agente de seguridad doméstica ○ Agente de entrenamiento físico ○ Agente de aprendizaje de idiomas
Multiagente	En los entornos multiagente, están presentes dos o más agentes que interactúan entre sí, pudiendo competir, cooperar o una combinación de ambos. Estos agentes perseguen objetivos y acciones independientes, pudiendo generar dinámicas complejas y estratégicas en el entorno. En estos entornos, los agentes pueden tener diferentes niveles de	○ Tráfico inteligente ○ Simulaciones de poblaciones ○ Robot que juega al futbol ○ Ayudante de matemáticas. ○ Juego del veintiuno (Black jack) ○ Juegos en línea

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	conocimiento, habilidades y capacidades, que influyan en su comportamiento y toma de decisiones. Los agentes pueden estar equipados con sensores para percibir el entorno y actuadores para realizar acciones que afecten al entorno. La interacción entre los agentes puede ser directa o indirecta, y puede involucrar comunicación, negociación, coordinación o competencia. El comportamiento de los agentes puede ser determinista o estocástico, evolucionado a lo largo del tiempo a medida que los agentes aprenden y se adaptan al entorno.	○ Sistemas de negociación electrónica ○ juegos competitivos ○ Sistemas de recomendación colaborativa ○ Sistemas de gestión del tráfico aéreo ○ Sistemas de coordinación de robots en almacenes ○ Juegos de estrategia en tiempo real en línea

Elaborado por los Autores

1.7.2 Clases de agentes

A continuación, se describen algunos de los agentes utilizados en inteligencia artificial:

Agentes inteligentes: Han transformado el diseño de las interfaces de usuario inteligentes, cuyo objetivo es reducir la sobrecarga de información del usuario. Para lograrlo, utilizan técnicas de filtrado de información basadas en el aprendizaje de las necesidades del usuario y las circunstancias en las que se realizan determinadas acciones (Brookshear & Brylow, 2020). Por ejemplo, en las interfaces de correo electrónico, los agentes pueden notificar al usuario la llegada de un nuevo correo que le interesa, evitando la distracción de en mensajes no deseados.

Las interfaces inteligentes aplican algoritmos de aprendizaje automático y procesamiento de lenguaje natural en el análisis del contenido de los mensajes, en la identificación de su relevancia y la presentación al usuario de únicamente la información más importante y pertinente. De esta manera, se mejora la eficiencia y la experiencia del usuario, al evitar la necesidad de revisar manualmente grandes volúmenes de información.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Agentes sintéticos: Estos agentes han revolucionado los entornos de realidad virtual, especialmente en la industria del entretenimiento, como películas, programas de televisión y publicidad, permitiendo generar imágenes por computadora que evitan la necesidad de construir maquetas complicadas para efectos especiales o de alquilar grandes cantidades de vestuario para escenas con multitudes.

La inteligencia artificial y las técnicas de generación de gráficos por computadora permiten simular personajes y objetos virtuales de manera realista, adaptándose a las necesidades de cada producción. La capacidad de trabajo autónomo que realizan estos agentes agiliza el proceso de producción y reduce los costos asociados.

Ofrecen flexibilidad creativa al permitir la generación de mundos virtuales detallados y escenas impactantes que antes serían difíciles o costosas de lograr, han abierto nuevas posibilidades en la creación de efectos visuales y en la representación de situaciones complejas.

Agentes conversacionales: Llamados también Chatterbots son programas diseñados para llevar a cabo conversaciones de forma auditiva o textual. Estos agentes encuentran aplicaciones en diversos campos, como el servicio al cliente y la adquisición de información. Su objetivo principal es simular la interacción humana y brindar respuestas relevantes y coherentes a través de la identificación de patrones en el lenguaje.

Los primeros chatterbots fuero Eliza (1966) y Parry (1972) permitían mantener conversaciones básicas con los usuarios. En la actualidad debido a los avances en inteligencia artificial y procesamiento del lenguaje natural, los modernos chatterbots son capaces de comprender el contexto, analizar el tono y el significado de las palabras, e incluso aprender de las interacciones previas para mejorar sus respuestas.

Los Chatterbots son ampliamente utilizados en aplicaciones de atención al cliente, asistentes virtuales y plataformas de chat en línea. Su capacidad para mantener conversaciones coherentes y proporcionar información relevante ha demostrado ser una herramienta valiosa en la automatización de tareas y en la mejora de la experiencia del usuario.

El lenguaje de marcas de Inteligencia Artificial (AIML - Artificial Intelligence Markup Language) es un intérprete que se utiliza para construir agentes conversacionales. AIML está especialmente diseñado para la creación de agentes de software con capacidad de comprensión y generación de lenguaje natural, conocidos como Alicebots. Estos

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

agentes son capaces de mantener conversaciones con usuarios y proporcionar respuestas adecuadas en función de los patrones de entrada definidos en las categorías.

Agentes móviles: Son agentes que tienen la capacidad de moverse o migrar de un host a otro en un entorno distribuido, poseen autonomía, tienen la capacidad de aprender, de percibir su entorno y responder de forma oportuna (Eze, 2022). Estos agentes pueden ejecutarse en diferentes máquinas sin necesidad de llevar consigo su propio código, lo que les permite aprovechar recursos distribuidos de manera eficiente.

Los agentes móviles son capaces de gestionar grandes volúmenes de información y facilitar la comunicación en tiempo real, esto hace que sean adecuados para aplicaciones que requieren interacciones dinámicas y respuestas rápidas. Estos agentes se utilizan en diversos contextos, como: supervisión y optimización de tráfico en las redes, sistemas de monitoreo y control, optimización de recursos, distribución de tareas, aplicaciones de colaboración, motores de búsqueda, asistentes personales, aprendizaje en la nube.

Agentes reactivos simples: Estos agentes toman decisiones basándose únicamente en la percepción actual del entorno, sin mantener un estado interno. Su comportamiento se basa en respuestas directas a los estímulos del entorno en tiempo real, sin considerar el pasado ni planificar el futuro de manera explícita. Estos agentes se centran en generar respuestas rápidas y efectivas a situaciones presentes, sin tener en cuenta el historial.

Estos agentes son adecuados en casos donde el estado actual del entorno proporciona suficiente información para tomar decisiones inmediatas. Son eficientes en términos de tiempo de respuesta y recursos computacionales, debido a que no requieren mantener un modelo interno ni realizar un procesamiento exhaustivo de la información.

Ejemplos de agentes reactivos simples son: un termostato inteligente que toma decisiones basándose únicamente en la información actual de la temperatura ambiente, un robot seguidor de línea que utiliza sensores para detectar una línea en el suelo y tomar decisiones en tiempo real para seguir la línea.

Agentes reactivos basados en modelos: Estos agentes mantienen un modelo interno del mundo que utilizan para tomar decisiones. Este modelo es una representación interna del estado del entorno y cómo evoluciona en respuesta a las acciones del agente.

Estos agentes tienen la capacidad de anticipar los efectos de sus acciones y planificar a corto plazo. Utilizan su modelo interno para simular diferentes secuencias de acciones posibles y evaluar los resultados esperados antes de tomar una decisión, esto les

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

permite considerar las consecuencias a corto plazo y seleccionar la acción que mejor se alinee con sus objetivos o criterios de rendimiento.

Los agentes reactivos basados en modelos son especialmente útiles en entornos donde es necesario planificar y tomar decisiones considerando las consecuencias futuras. Por ejemplo, en un juego de ajedrez, el agente puede simular diferentes movimientos y evaluar cómo afectarían al tablero y a su posición futura. Esto les permite tomar decisiones estratégicas y anticiparse a las jugadas del oponente (Knott, 2017).

Agentes basados en objetivos: Estos agentes tienen metas u objetivos bien claros de lo que desean alcanzar en su entorno. Se guían por la evaluación de su estado actual en relación con dichos objetivos y toman acciones que los acerquen a ellos.

Estos agentes mantienen un modelo interno del estado del entorno y utilizan este modelo para evaluar qué tan cerca están de alcanzar sus metas. Analizan las posibles acciones disponibles y seleccionan la que mejor se alinee con sus objetivos. Esta selección se basa en algún tipo de función de evaluación que considera factores como el costo, la utilidad o la probabilidad de éxito.

A medida que el agente toma acciones y su estado cambia, vuelven a evaluar su posición en relación con los objetivos y ajustan sus decisiones. También pueden considerar diferentes opciones y seleccionar la que maximice la probabilidad de alcanzar sus metas (MacKay, 2003).

Los agentes basados en objetivos son muy útiles en entornos en los que se establecen metas claras y los caminos para lograrlas pueden ser diversos. Por ejemplo, en un sistema de planificación de rutas, un agente basado en objetivos si tiene como objetivo llegar a un destino específico debe considerar diferentes rutas posibles en función del tiempo, la distancia o las restricciones de tráfico.

Agentes basados en utilidad: Estos agentes evalúan las acciones posibles en términos de su utilidad o valor esperado, tomando decisiones que maximizan dicha utilidad.

La utilidad representa la medida de satisfacción o beneficio que el agente espera obtener al tomar una determinada acción en un estado dado. Cada acción posible se evalúa en función de la utilidad esperada, que se ha estimado que proporcionará al agente.

Estos agentes de inteligencia artificial no se limitan a evaluar únicamente los resultados inmediatos de sus acciones; también consideran cuidadosamente las

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

implicaciones a largo plazo y las probabilidades asociadas a cada posible resultado (Das, 2008). Para ello, aplican modelos avanzados o funciones de utilidad que les permiten calcular la utilidad esperada de cada opción de acción, seleccionando aquella que maximice su valor global.

La función de utilidad incorpora una variedad de factores, incluyendo los objetivos específicos del agente, los costes asociados a cada acción, los riesgos e incertidumbres presentes en el entorno, así como las preferencias propias del agente. Utilizando esta función, el agente es capaz de asignar valores numéricos a diferentes acciones y realizar comparaciones objetivas entre ellas, lo que le facilita la toma de la decisión más óptima y estratégica dentro del contexto en el que trabaja.

Un ejemplo de aplicación de agentes basados en utilidad es en sistemas de recomendación, donde el agente busca maximizar la utilidad del usuario ofreciendo recomendaciones personalizadas que se ajusten a sus preferencias y necesidades.

Agentes basados en conocimiento: Estos agentes utilizan conocimiento experto o reglas específicas para tomar decisiones, basándose en una base de conocimientos previamente definida. Se trata de agentes que aplican razonamiento lógico y utilizan reglas o algoritmos específicos para resolver problemas o responder preguntas.

El conocimiento experto proveniente de expertos humanos en un dominio específico se captura y codifica en forma de reglas o declaraciones lógicas. Este conocimiento incluye hechos, reglas, relaciones y restricciones relevantes para el dominio en cuestión.

Los agentes basados en conocimiento utilizan su acervo de información especializada para realizar inferencias complejas y tomar decisiones informadas. Emplean técnicas sofisticadas como la lógica proposicional, la lógica de predicados o sistemas avanzados basados en reglas para procesar y razonar sobre el conocimiento acumulado, conduciéndoles a conclusiones lógicas y fundamentadas.

El proceso de toma de decisiones en estos agentes implica una búsqueda meticulosa en su base de conocimientos para identificar reglas o hechos pertinentes a la situación específica. Luego, aplican razonamiento lógico detallado para derivar conclusiones coherentes. Estas conclusiones son trascendentales, debido a que guían al agente en la selección de acciones más adecuadas o en la generación de respuestas precisas a los desafíos que enfrentan.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Los agentes basados en conocimiento son especialmente útiles en dominios en los que se dispone de un conocimiento experto bien definido y estructurado. Se utilizan en sistemas de diagnóstico médico, asistentes virtuales, sistemas de recomendación o en cualquier otro contexto en el que se requiera un razonamiento lógico basado en conocimiento específico.

Agentes multiagente: Son sistemas compuestos por varios agentes individuales que interactúan y colaboran entre sí para lograr un objetivo común o resolver un problema de manera cooperativa. Cada agente en el sistema es autónomo y puede tener diferentes características, habilidades y conocimientos (Eze, 2022).

La interacción entre los agentes se llevarse a cabo a través de diferentes mecanismos, como la comunicación directa, el intercambio de información, la negociación y la coordinación. Los agentes comparten conocimientos, realizan tareas específicas, cooperan en la toma de decisiones y coordinan acciones para alcanzar un objetivo conjunto.

Los sistemas multiagente son utilizados en una amplia gama de aplicaciones, como la logística, el tráfico inteligente, la gestión de recursos, la robótica colaborativa y los juegos en línea, entre otros. En estas aplicaciones, los agentes trabajan juntos para abordar problemas complejos que requieren la colaboración de múltiples entidades.

A***gentes basados en aprendizaje:*** Representan una categoría avanzada de sistemas inteligentes, dotados de la capacidad de auto-mejora y adaptación a través de la experiencia. Estos agentes incorporan algoritmos de aprendizaje automático de última generación para procesar y aprender de los datos obtenidos de su entorno.

El proceso de aprendizaje en estos agentes es sofisticado e implica no solo el reconocimiento de patrones y la identificación de relaciones, sino también la extracción y análisis profundo de información pertinente de los datos de entrada. Aprenden mediante un proceso continuo de retroalimentación, donde comparan los resultados actuales con los objetivos o resultados ideales, y ajustan sus métodos y estrategias en consecuencia.

Con el tiempo, estos agentes evolucionan y optimizan su comportamiento, tomando decisiones cada vez más precisas y efectivas basadas en el cúmulo de experiencias pasadas y el conocimiento acumulado. Esta capacidad de adaptación y aprendizaje continuo los hace extraordinariamente eficaces en una amplia variedad de entornos y aplicaciones.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Existen diversos métodos de aprendizaje utilizados por agentes basados en el aprendizaje, cada uno con características y aplicaciones distintas. En el aprendizaje supervisado, los agentes trabajan con conjuntos de datos etiquetados, aprendiendo a clasificar o predecir nuevos datos basándose en ejemplos previos (Xiao, 2022). Por otro lado, en el aprendizaje no supervisado, los agentes exploran datos sin etiquetar para descubrir patrones y estructuras ocultas por sí mismos. Mientras tanto, en el aprendizaje por refuerzo, los agentes mejoran su rendimiento interactuando directamente con el entorno, siendo recompensados o penalizados según las consecuencias de sus acciones.

Los agentes basados en aprendizaje tienen la notable capacidad de adaptarse y perfeccionar su desempeño con la exposición a más datos y experiencias. Estos sistemas ajustan continuamente sus modelos internos y estrategias, optimizando su comportamiento para alcanzar los objetivos.

Estos agentes encuentran aplicación en una amplia gama de campos, incluyendo reconocimiento de voz, visión por computadora, recomendación de productos, análisis de grandes conjuntos de datos y toma de decisiones en tiempo real. Mediante el uso del aprendizaje automático, estos agentes pueden procesar y aprender eficientemente de grandes volúmenes de datos, adaptándose a entornos y requerimientos en constante cambio.

Agentes basados en reglas: Son sistemas inteligentes que utilizan un conjunto de reglas predefinidas para tomar decisiones y realizar acciones en función de la información disponible. Estas reglas pueden estar basadas en conocimiento experto o en reglas lógicas, y son diseñadas por expertos en el dominio específico del problema.

El funcionamiento de estos agentes se basa en la evaluación de condiciones y la aplicación de acciones correspondientes. Cada regla consta de una condición o conjunto de condiciones que deben ser satisfechas para que la regla se aplique, y una acción que se llevará a cabo si se cumplen las condiciones.

Cuando un agente recibe información del entorno, evalúa las reglas una por una en orden y verifica si las condiciones de cada regla se cumplen. Si encuentra una regla cuyas condiciones se satisfacen, el agente ejecuta la acción asociada a esa regla. Si ninguna regla se cumple, el agente puede tener una regla predeterminada o una acción por defecto a realizar.

Los agentes basados en reglas son especialmente útiles cuando el dominio del

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

problema es bien estructurado y se pueden identificar claramente las reglas que guían el comportamiento del agente. Estos agentes son utilizados en sistemas expertos, donde el conocimiento de un experto en un campo específico se codifica en forma de reglas para realizar tareas de diagnóstico, asesoramiento o toma de decisiones.

Agentes basados en comportamiento: Estos sistemas inteligentes son diseñados para emular el comportamiento de seres vivos, ya sea humano u otros organismos. Estos agentes se centran en replicar acciones y respuestas específicas en función de estímulos del entorno, con el objetivo de generar un comportamiento realista y coherente.

El enfoque principal de estos agentes es la simulación de patrones de comportamiento observados en seres vivos. Estos patrones pueden incluir movimientos, reacciones a estímulos, toma de decisiones y acciones que se asemejan a las de organismos reales. Para lograrlo, estos agentes suelen utilizar técnicas como los algoritmos de inteligencia artificial y el modelado de sistemas complejos.

Los agentes basados en comportamiento son comunes en simulaciones y juegos, donde se busca recrear entornos virtuales con comportamientos realistas. Por ejemplo: juegos de simulación de vida, juegos de simulación de ciudades o de mascotas virtuales; los agentes basados en comportamiento pueden imitar acciones y respuestas similares a las de los seres vivos para proporcionar una experiencia de juego más inmersiva.

Estos agentes también se utilizan en aplicaciones de entrenamiento y aprendizaje, donde se busca replicar comportamientos específicos para fines educativos o de entrenamiento. Por ejemplo, en simuladores de vuelo, los agentes basados en comportamiento pueden replicar las acciones y reacciones de pilotos reales para enseñar a los usuarios cómo operar una aeronave.

Agentes híbridos: Estos agentes son sistemas inteligentes que combinan características y técnicas de diferentes tipos de agentes que se mencionaron anteriormente. Estos agentes aprovechan las fortalezas de múltiples enfoques para lograr un desempeño óptimo en una variedad de situaciones.

La idea detrás de los agentes híbridos es capitalizar las ventajas de diferentes paradigmas de agente para abordar de manera efectiva problemas complejos y dinámicos. Por ejemplo, un agente híbrido puede combinar elementos de agentes reactivos y basados en conocimiento. Esto significa que el agente toma decisiones rápidas y eficientes en situaciones conocidas utilizando reglas o conocimiento predefinido, y también puede

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

adaptarse y aprender de nuevas situaciones utilizando técnicas de aprendizaje automático.

Otro ejemplo es un asistente virtual en un dispositivo móvil. Este agente puede utilizar un enfoque reactivo para responder rápidamente a comandos de voz o preguntas frecuentes, aprovechando patrones y reglas predefinidas. Al mismo tiempo, puede incorporar técnicas de aprendizaje automático para mejorar su capacidad de entender el contexto del usuario y brindar respuestas más personalizadas y precisas.

Cada tipo de agente tiene sus propias fortalezas y limitaciones, y la elección del tipo de agente adecuado dependerá del problema o la tarea específica que se esté abordando. La selección del tipo de agente correcto es fundamental para lograr un rendimiento óptimo en sistemas de Inteligencia Artificial.

1.7.3 Arquitecturas de programas para agentes

Estas arquitecturas se basan en los objetivos aplicativos del agente. Las arquitecturas de agentes a ser revisadas son: arquitectura reactiva simple, arquitectura deliberante, arquitectura por finalidad, arquitectura basada en el razonamiento humano, arquitectura de agente de reforzamiento, arquitectura de agente de aprendizaje profundo, arquitectura de agente multiagente.

a) Arquitectura reactiva simple

La arquitectura reactiva simple es una forma básica de diseño de agentes en la que el comportamiento del agente se basa en una correspondencia directa entre el estímulo recibido y la respuesta generada. En esta arquitectura, el agente no posee habilidades de toma de decisiones sofisticadas, sino que simplemente reacciona al entorno en el que se encuentra. El agente percibe el entorno a través de sensores y, en función de esa percepción, elige la acción más apropiada para interactuar con el entorno. Esta elección de acción se realiza mediante reglas de condición-acción (si-entonces), que establecen la conexión entre la percepción actual y la acción a tomar.

Esta arquitectura es adecuada para entornos simples donde las percepciones históricas no son relevantes y solo se requiere una respuesta inmediata y directa al estímulo presente. Puede ser implementada en hardware o mediante software con algoritmos de búsqueda rápidos.

Es importante destacar que esta arquitectura solo funciona correctamente en entornos totalmente observables, es decir, cuando el agente tiene acceso a toda la información relevante del entorno. Si hay partes del entorno que no son observables,

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

pueden surgir problemas debido a la falta de información completa para tomar decisiones precisas.

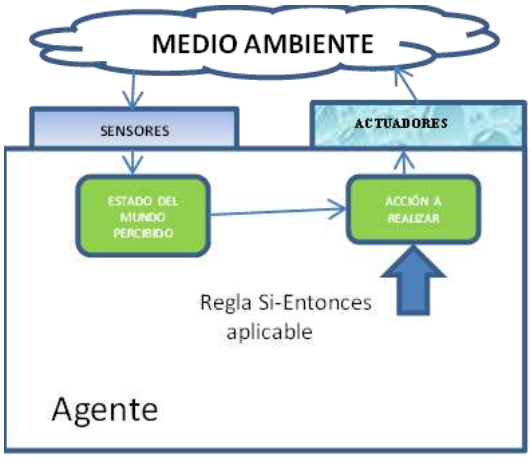


Figura 1.8 Arquitectura Reactiva Simple

Elaborado por los Autores

La figura 1.8 representa de manera esquemática un agente que utiliza la arquitectura reactiva simple. Este agente está equipado con sensores que le permiten percibir el estado del mundo en el que trabaja, también cuenta con actuadores que le permiten llevar a cabo acciones específicas en respuesta a esa percepción.

El agente utiliza los sensores para recopilar información sobre el entorno y generar una representación interna del estado actual del mundo. Basándose en esta información, el agente consulta las reglas de condición-acción predefinidas que tiene almacenadas. Estas reglas establecen condiciones específicas, que cuando se cumplen, indican qué acción debe tomar el agente. Una vez que el agente determina la acción adecuada en función de las reglas de condición-acción, utiliza los actuadores para llevar a cabo esa acción en el entorno. Estas acciones pueden incluir manipulación física, interacción con otros agentes o cualquier otro tipo de respuesta que sea necesaria en ese contexto.

b) Arquitectura deliberante

La arquitectura deliberante, representada en la figura 1.9, es una estructura más compleja que la arquitectura reactiva simple. Esta arquitectura es adecuada para entornos parcialmente observables, donde el agente necesita tener en cuenta un historial reciente de información para tomar decisiones.

A diferencia de la arquitectura reactiva simple, el agente deliberante tiene la capacidad de deliberar sobre la acción a realizar en función de la información disponible

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

y el estado actual del entorno. Para esto, el agente recopila y analiza tanto la información sobre la evolución del entorno como las posibles consecuencias de sus acciones.

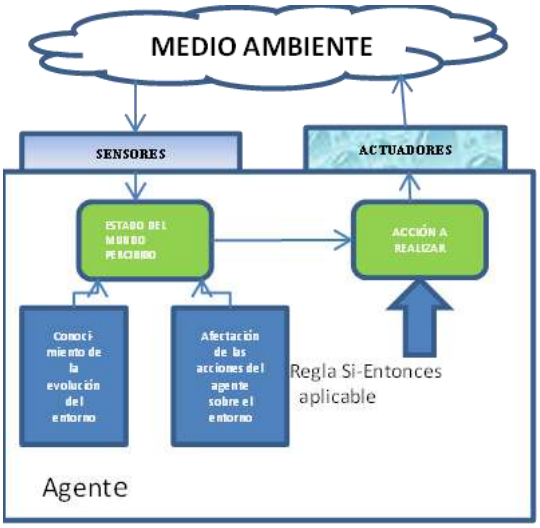


Figura 1.9 Arquitectura Deliberante

Elaborado por los Autores

El proceso de selección de acción en un agente de IA puede implementarse a través de reglas de producción o mediante el empleo de redes neuronales, dependiendo de la complejidad del problema y la disponibilidad de información. Las reglas de producción proporcionan un marco lógico y estructurado para evaluar opciones y tomar decisiones, mientras que las redes neuronales, con su capacidad de aprendizaje profundo y adaptativo, permiten al agente analizar y procesar una gama más amplia y compleja de datos para seleccionar la acción más pertinente que le permita alcanzar sus objetivos.

La arquitectura deliberante, es particularmente valiosa en el manejo de problemas complejos y en la realización de tareas de planificación avanzada. En esta arquitectura, el agente es capaz de ejecutar secuencias de acciones de forma estratégica y calculada, con el fin de lograr objetivos específicos dentro de su entorno. Esta capacidad para deliberar y planificar a largo plazo es un aspecto importante en el diseño de agentes inteligentes destinados a resolver tareas que requieren un nivel superior de cognición y adaptación (Gelfond & Kahl, 2014).

c) Arquitectura por finalidad

La arquitectura por finalidad, representada en la figura 1.10, es una estructura que se basa en la idea de que el agente debe tener una comprensión clara de su objetivo final para tomar decisiones efectivas en cada estado del entorno. Además de la percepción del mundo en cada nuevo estado, es necesario tener información sobre la dirección en la que

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

el agente desea llegar, de modo que pueda evaluar si los estados intermedios alcanzados por las acciones tomadas lo acercarán o alejarán de su objetivo final.

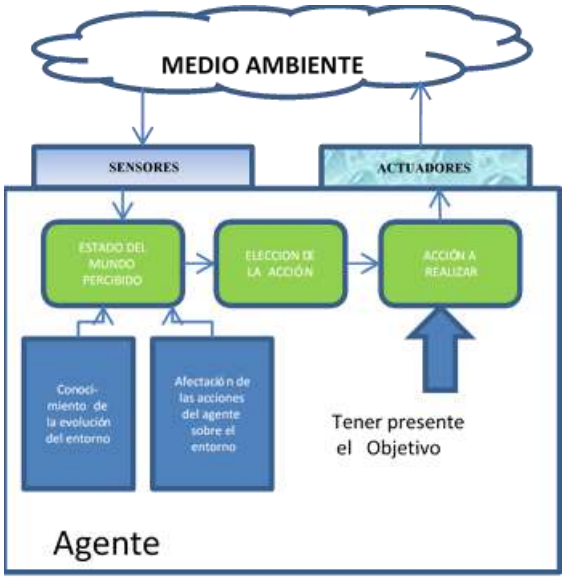


Figura 1.10 Arquitectura por Finalidad

Elaborado por los Autores

En esta arquitectura, el agente tiene flexibilidad para modificar su conocimiento y adaptarse en función de las decisiones que va tomando. El estado actual del entorno se actualiza constantemente a través de los sensores del agente, y cuando se identifica una acción que se puede realizar, el agente la lleva a cabo mediante el control de sus actuadores.

Las estrategias de búsqueda son un ejemplo concreto de esta arquitectura. El agente utiliza técnicas de búsqueda para explorar el espacio de posibles acciones y encontrar una secuencia de acciones que lo lleve hacia su objetivo final. A medida que el agente recopila información a través de la percepción y realiza acciones, puede evaluar si se está acercando o alejando de su objetivo, y ajustar su estrategia en consecuencia.

d) Arquitectura basada en el razonamiento humano

La arquitectura que se muestra en la figura 1.11 es un ejemplo de una arquitectura de agente basada en el razonamiento humano. En este caso, el agente opera en un entorno parcialmente observable, lo que significa que no tiene acceso completo a toda la información del entorno. Sin embargo, el agente es capaz de formarse una idea del estado del medio ambiente en el que existe utilizando la información disponible a través de sus sensores.

En esta arquitectura, el agente tiene objetivos que actúan como su motivación

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

principal para alcanzar una meta deseada. Para lograr esto, el agente necesita seleccionar una o más acciones que modificarán el estado del medio ambiente con el fin de satisfacer su necesidad de alcanzar la meta. Esta toma de decisiones se realiza considerando los objetivos y evaluando cómo cada acción contribuirá a su logro.

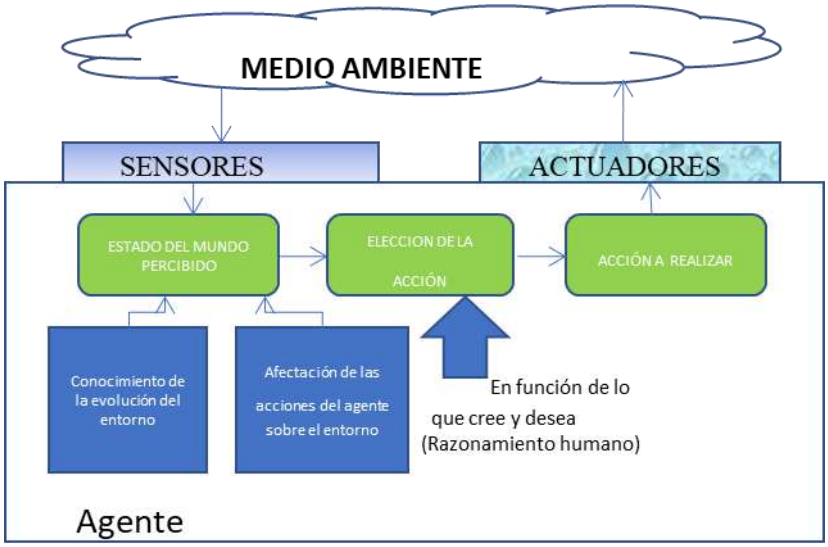


Figura 1.11 Arquitectura Basado en el razonamiento humano

Elaborado por los Autores

La arquitectura es especialmente útil cuando existen objetivos conflictivos y se requiere buscar un equilibrio entre ellos para alcanzar la meta deseada. También es adecuada cuando hay múltiples objetivos y existe incertidumbre sobre la posibilidad de alcanzarlos todos. En tales casos, el agente necesita evaluar las probabilidades de éxito de cada objetivo y tomar decisiones que maximicen las posibilidades de alcanzar la meta general.

e) Arquitectura de agente de reforzamiento

El agente por refuerzo emplea una técnica de aprendizaje automático que permite a un agente aprender a tomar decisiones secuenciales óptimas a través de la interacción con su medio ambiente. Las decisiones óptimas el agente las toma mediante la maximización de una recompensa numérica. El agente interactúa con su medio ambiente y recibe una señal de recompensa cada vez que realiza una acción.

El agente no se guía mediante instrucciones explícitas sobre las acciones a tomar, en su lugar, adquiere conocimiento a través de la experiencia y el error. Cada vez que el agente realiza una acción, el entorno le otorga una recompensa numérica. La meta del

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

agente es maximizar la acumulación de estas recompensas a lo largo del tiempo.

En el aprendizaje por refuerzo, surge un desafío fundamental: encontrar el equilibrio adecuado entre la exploración de nuevas acciones y la explotación de aquellas acciones que han demostrado producir recompensas significativas. El agente (figura 1.12) debe aprender a explorar lo suficiente para descubrir estrategias efectivas y luego aprovechar esas estrategias para maximizar sus recompensas.

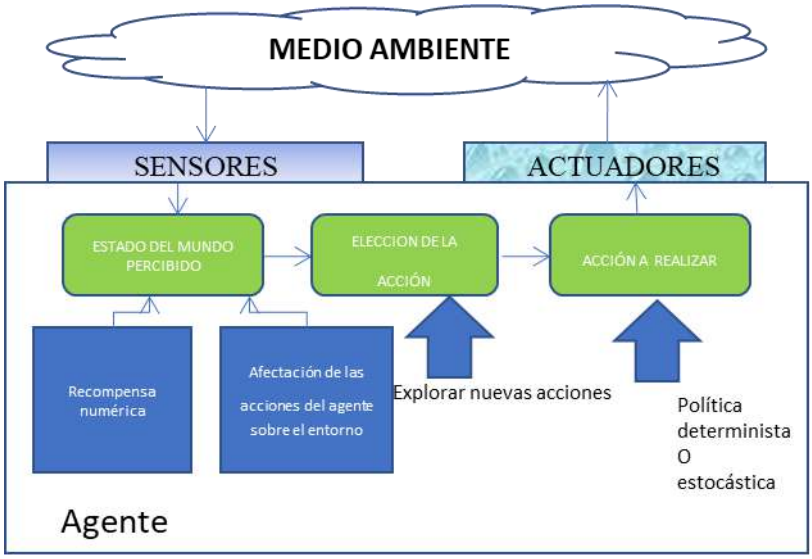


Figura 1.12 Arquitectura de agente de reforzamiento

Elaborado por los Autores

El agente utiliza una política para determinar qué acción tomar en función de su estado actual y de su conocimiento. La política puede ser determinista o estocástica, dependiendo de si siempre elige la misma acción en un estado dado o si toma decisiones basadas en probabilidades.

El núcleo fundamental del aprendizaje por refuerzo radica en la habilidad de aprender de manera progresiva a largo tiempo, a medida que se actualizan valores denominados valores de acción o funciones de valor. Estos valores proporcionan una estimación de la utilidad prevista al ejecutar una acción específica en un estado particular.

Dentro del campo del aprendizaje por refuerzo, se encuentran diversos algoritmos destacados, como Q-learning, SARSA, DDPG (Deep Deterministic Policy Gradients) y A3C (Asynchronous Advantage Actor-Critic). Cada uno de estos algoritmos se especializa en abordar aspectos específicos del aprendizaje por refuerzo, como la exploración, la estimación de valores y política de actualización.

f) Arquitectura de agente de aprendizaje profundo

Esta arquitectura (figura 1.13) representa una categoría avanzada de enfoques en

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

la inteligencia artificial, fundamentada en el uso de redes neuronales profundas o DNN (Deep Neural Networks). Estas redes, compuestas por múltiples capas de neuronas artificiales interconectadas, están diseñadas para aprender de grandes volúmenes de datos y realizar tareas específicas de manera autónoma (Haykin, 2009). Imitando la complejidad del cerebro humano, estas redes permiten a las máquinas identificar patrones, realizar predicciones y resolver problemas complejos sin necesidad de intervención humana directa. El aprendizaje profundo, impulsado por estas redes, ha revolucionado campos como el procesamiento de lenguaje natural, la visión por computadora, la generación de texto, los videojuegos, la traducción automática, la robótica, la toma de decisiones y las simulaciones avanzadas.

Una Red Neuronal Profunda se estructura en múltiples capas de neuronas artificiales. Cada capa consiste en un conjunto de neuronas, también conocidas como nodos o unidades, y se clasifican generalmente en tres categorías: la capa de entrada, una o varias capas ocultas, y la capa de salida. La profundidad de estas redes se define por el número de capas ocultas presentes entre la entrada y la salida. A mayor número de capas ocultas, más 'profunda' se considera la red, lo que implica una mayor capacidad para aprender características complejas y abstractas de los datos.

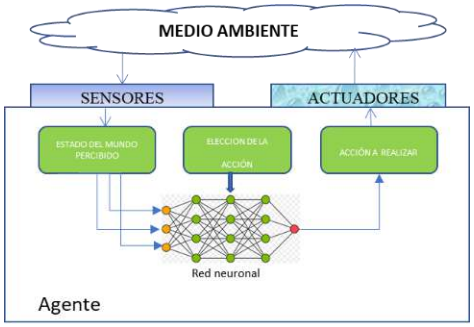


Figura 1.13 Arquitectura de agente de aprendizaje profundo

Elaborado por los Autores

La profundidad en las redes neuronales profundas permite una representación jerárquica de los datos. A medida que los datos pasan a través de las capas, las características y abstracciones se vuelven progresivamente más complejas y abstractas. Esto significa que la red puede aprender automáticamente características simples en las primeras capas, como bordes y texturas, y luego construir representaciones más abstractas, como formas y objetos, en capas posteriores.

Entrenar redes neuronales profundas puede ser un desafío computacional, debido a que la retropropagación de gradientes (el algoritmo utilizado para ajustar los pesos de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

la red) puede volverse inestable en redes profundas (Kowaliw et al., 2014). Sin embargo, se han desarrollado técnicas como la normalización de lotes, funciones de activación apropiadas y optimizadores avanzados para abordar estos desafíos y permitir el entrenamiento eficiente.

Además de las redes neuronales estándar existen las Redes Neuronales Convolucionales (CNN) que son excelentes para el procesamiento de datos estructurados a través de matrices bidimensionales, como la clasificación y segmentación de imágenes y las Redes Neuronales Recurrentes (RNN) que son efectivas para datos secuenciales, como el procesamiento de lenguaje natural (Haykin, 2009).

El Aprendizaje por Reforzamiento Profundo (DRL) es un subcampo esencial en el que los agentes emplean redes neuronales profundas para aprender a tomar decisiones secuenciales en entornos interactivos, esto se logra a través de la retroalimentación de recompensas, donde la red es entrenada con el objetivo de maximizar las recompensas acumuladas a lo largo del tiempo.

g) Arquitectura de agente multiagente

La arquitectura multiagente (figura 1.14) representa un enfoque innovador y fascinante en el campo de la inteligencia artificial y la robótica. Este enfoque se centra en el diseño y la implementación de sistemas en los que múltiples agentes, ya sean entidades autónomas como robots, programas de software o incluso seres humanos, colaboran para alcanzar objetivos tanto individuales como colectivos (Franceschetti, 2018). A través de su interacción y cooperación, estos agentes pueden abordar problemas y ejecutar tareas con mayor eficacia que si operaran de forma aislada.

Esta arquitectura destaca por su potencial para crear sistemas inteligentes altamente adaptativos, capaces de funcionar eficientemente en entornos cambiantes y bajo condiciones complejas y dinámicas. Entre sus aplicaciones prácticas más relevantes se encuentran el desarrollo de vehículos autónomos, la robótica avanzada, la planificación logística, los sistemas de transporte inteligentes, la gestión de tráfico en ciudades inteligentes, la simulación de sistemas complejos y el desarrollo de modelos financieros sofisticados.

Cada agente dentro de una arquitectura multiagente opera como una entidad autónoma, equipado con sus propias metas y capacidades de toma de decisiones. Esto permite que cada agente tome decisiones de manera independiente, basándose en su

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

conocimiento particular y objetivos individuales, contribuyendo así a la eficacia y adaptabilidad del sistema en su conjunto.

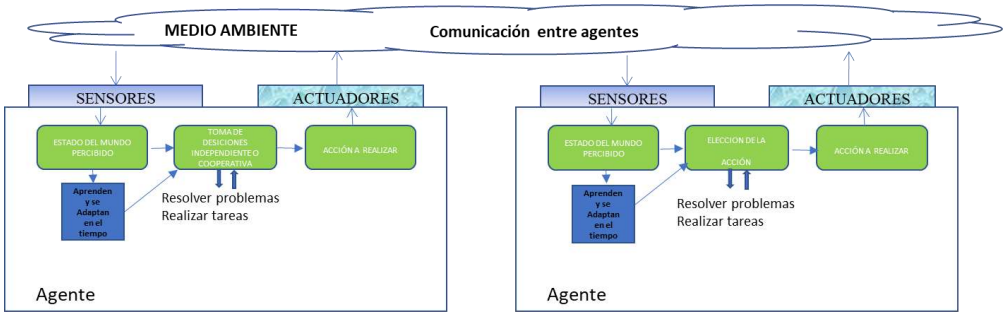


Figura 1.14 Arquitectura de agente multiagente

Elaborado por los Autores

Teniendo los agentes objetivos individuales pueden al mismo tiempo trabajar juntos para lograr objetivos colectivos. Esta dualidad permite una amplia gama de aplicaciones, desde sistemas de control de tráfico aéreo, donde los aviones individuales buscan sus rutas óptimas mientras evitan colisiones, hasta sistemas de recomendación en línea, donde los agentes (en calidad de algoritmos de recomendación) trabajan juntos para brindar recomendaciones personalizadas a los usuarios.

La comunicación entre agentes es esencial, pueden comunicarse directa o indirectamente para compartir información relevante, coordinar acciones y tomar decisiones informadas. La comunicación puede ser unidireccional o bidireccional y puede implicar intercambios de datos, mensajes, señales, eventos o incluso lenguaje natural en sistemas más avanzados. La elección de cómo se implementa la comunicación entre agentes depende de la aplicación específica y de los requisitos del sistema. En muchas arquitecturas multiagente, se utilizan múltiples formas de comunicación en combinación para permitir una interacción efectiva entre los agentes. La comunicación efectiva es esencial para lograr la colaboración y la coordinación entre los agentes y para alcanzar los objetivos individuales o colectivos en un sistema multiagente.

En una arquitectura multiagente, la toma de decisiones distribuida es una característica clave, pueden tomar decisiones de manera independiente y cooperativa. Esto a menudo involucra la coordinación de acciones para evitar conflictos o redundancias y para lograr objetivos comunes (Shi, 2019).

Los agentes pueden aprender y adaptarse a lo largo del tiempo, utilizando técnicas de aprendizaje automático y algoritmos de refuerzo para mejorar sus estrategias y tomar

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

decisiones más efectivas. Este aprendizaje puede ser individual o colectivo, donde los agentes comparten conocimientos.

Estas arquitecturas son adecuadas para modelar entornos dinámicos donde las situaciones cambian constantemente. Los agentes deben adaptarse y reaccionar en tiempo real a nuevas circunstancias.

La escalabilidad es un reto importante en sistemas con un gran número de agentes, la arquitectura debe ser eficiente para manejar múltiples agentes y garantizar un rendimiento adecuado.

1.8 El conocimiento

El razonamiento humano, que se fundamenta en el conocimiento acumulado para derivar conclusiones, sirve como base para el desarrollo de sistemas en el campo de la Inteligencia Artificial. En IA, la investigación se puede dividir en dos corrientes principales: la inteligencia simbólica y la inteligencia computacional.

La inteligencia simbólica, también conocida como IA tradicional, se enfoca en la resolución de problemas mediante un razonamiento basado en conocimientos explícitos. Esta vertiente utiliza representaciones simbólicas y reglas lógicas para procesar información y deducir soluciones. Este enfoque se caracteriza por la programación de algoritmos complejos y la manipulación de símbolos para emular tareas cognitivas.

Por otro lado, la inteligencia computacional aborda la resolución de problemas a partir de la conexión y análisis de datos. Esta corriente incluye técnicas avanzadas como redes neuronales artificiales, algoritmos genéticos, sistemas difusos (fuzzy logic), programación evolutiva y vida artificial (Martinsanz & Peñas, 2005). Estas técnicas se inspiran en el funcionamiento de sistemas biológicos y tienen como objetivo replicar su capacidad innata de aprendizaje y adaptación, permitiendo a los sistemas de IA mejorar y evolucionar de forma autónoma.

Ambas categorías de investigación en IA tienen sus propias fortalezas y aplicaciones. La inteligencia simbólica es efectiva para problemas que se pueden expresar en términos lógicos y que requieren razonamiento basado en reglas y conocimiento explícito. Por otro lado, la inteligencia computacional es útil en problemas donde los datos son complejos y se requiere un enfoque más basado en el aprendizaje y la adaptación.

La Inteligencia Artificial (IA) se puede entender como la ciencia que se ocupa de la ingeniería del conocimiento, abordando la representación, adquisición y aplicación del

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

mismo para resolver problemas reales. En este contexto, la IA actúa como un conjunto de ideas y enfoques acerca de cómo representar y utilizar el conocimiento, así como el desarrollo de sistemas informáticos que puedan aprovechar dicho conocimiento.

Desde un punto de vista práctico, la IA se centra en el diseño y construcción de sistemas inteligentes capaces de simular el pensamiento humano y tomar decisiones informadas. Esto implica desarrollar algoritmos, modelos y herramientas que permitan capturar el conocimiento de expertos en un dominio particular y utilizarlo para resolver tareas complejas.

Desde la perspectiva científica, la Inteligencia Artificial (IA) se dedica a explorar y explicar diversas formas de inteligencia a través de la representación del conocimiento y su aplicación en el desarrollo de sistemas informáticos.

La representación del conocimiento es un componente trascendente en el campo de la Inteligencia Artificial, debido a que se refiere a cómo se almacena y organiza la información esencial para realizar tareas específicas. Para lograr una representación eficaz, se utilizan diversas técnicas y herramientas, como estructuras de datos avanzadas, ontologías, bases de conocimiento y sistemas expertos, adaptándose al dominio específico y a los objetivos del sistema (Poole & Mackworth, 2010).

El éxito en la construcción de sistemas de IA depende en gran medida de desarrollar representaciones del conocimiento que sean no solo adecuadas sino también manipulables de manera eficiente. Una representación inadecuada impediría una gestión efectiva del conocimiento. Esto conlleva la necesidad de seleccionar métodos de codificación de la información que resulten comprensibles y manejables por el sistema de IA.

Una representación del conocimiento bien diseñada es vital para la capacidad de un sistema de IA de ejecutar tareas inteligentes, incluyendo el procesamiento de lenguaje natural, la toma de decisiones informadas, la planificación estratégica y la resolución de problemas complejos. Además, una representación bien estructurada es fundamental para facilitar el proceso de aprendizaje automático, permitiendo al sistema adquirir nuevo conocimiento y mejorar su desempeño a través de la experiencia.

La elección de la representación del conocimiento también está influenciada por la naturaleza del problema y el tipo de datos involucrados. Por ejemplo, en problemas de procesamiento de texto, se pueden utilizar técnicas basadas en lenguaje natural y modelos

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

de lenguaje para representar la información. En problemas de visión por computadora, se pueden emplear redes neuronales convolucionales para extraer características y representar las imágenes. Para tareas de diagnóstico médico suelen utilizarse sistemas expertos basados en reglas, mientras que para aplicaciones de procesamiento de lenguaje natural se prefieren representaciones distribucionales como embeddings o modelos de lenguaje.

La mayoría de la conducta inteligente se basa en una representación directa del conocimiento, para lo cual la lógica formal proporciona un enfoque importante. El conocimiento, en especial el llamado conocimiento común (analizar, conjeturar, pronosticar y decidir), es la base del comportamiento inteligente.

La Ciencia Cognitiva abarca el estudio de las percepciones humanas y el procesamiento mental de la información, desde la entrada sensorial hasta la resolución de problemas complejos. Esto incluye las actividades intelectuales de individuos y de la sociedad en su conjunto, además de la investigación sobre las características tanto de la inteligencia humana como de la inteligencia artificial (Shi, 1990).

Considerada una base teórica fundamental para la Inteligencia Artificial, la ciencia cognitiva es un campo interdisciplinario que se ha desarrollado a partir de la psicología moderna, ciencias de la Información, neurociencia, matemáticas, ciencia lingüística, antropología, filosofía natural y otras disciplinas relevantes. La convergencia de estos campos ha permitido una comprensión más profunda de los procesos mentales y cognitivos que subyacen a la inteligencia y la toma de decisiones, tanto en humanos como en sistemas de IA.

La ciencia cognitiva proporciona una base teórica sólida para el diseño y desarrollo de sistemas de IA que buscan emular el pensamiento y el comportamiento humano. Al combinar conocimientos sobre el funcionamiento del cerebro, la percepción, el lenguaje, el aprendizaje y la memoria, se puede mejorar la capacidad de los sistemas de IA para procesar información, aprender de la experiencia y tomar decisiones inteligentes.

En el ámbito de la IA, la ciencia cognitiva ha influido en la creación de modelos y algoritmos inspirados en el funcionamiento del cerebro y los procesos mentales humanos. Por ejemplo, las redes neuronales artificiales están basadas en la estructura y funcionamiento de las neuronas biológicas, y se utilizan para realizar tareas de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

aprendizaje y reconocimiento de patrones.

El conocimiento, que representa el conjunto de hechos e información acumulada por la humanidad, es un elemento clave en la modelación de los diversos aspectos del mundo que nos rodea. Su recopilación y procesamiento son esenciales para facilitar la toma de decisiones adecuadas y oportunas, impulsando a su vez el aprendizaje y la adquisición de nuevo conocimiento (Brachman & Levesque, 2004).

En el ámbito de la Inteligencia Artificial, el conocimiento adquiere una relevancia crítica. Los sistemas inteligentes se basan en una amplia gama de datos y conocimientos previos para ejecutar tareas complejas y tomar decisiones informadas. La adquisición y gestión eficaz del conocimiento son fundamentales para que los agentes de IA puedan aprender de sus experiencias y mejorar su rendimiento de manera continua.

Los progresos en el aprendizaje automático y la minería de datos han abierto nuevas puertas para los sistemas de IA, permitiéndoles acceder a vastas cantidades de información y extraer conocimientos valiosos. Modelos avanzados, como las redes neuronales, son capaces de analizar datos para descubrir patrones ocultos, lo que les permite aprender y adaptarse a nuevas circunstancias.

El proceso de aprendizaje en los sistemas de IA es paralelo al proceso de adquisición de conocimiento humano. Mediante la exposición a datos variados y la experiencia acumulada, los agentes de IA perfeccionan su rendimiento y alcanzan una mayor precisión en sus predicciones y decisiones.

El conocimiento puede ser clasificado en tres formas:

1) **Conocimiento declarativo:** El conocimiento declarativo es aquel que se almacena en la memoria en forma de hechos, conceptos e ideas estáticas, expresadas mediante sentencias o proposiciones acerca del mundo que nos rodea. Estos conocimientos se presentan como patrones que permiten encontrar relaciones entre fragmentos de información. Un ejemplo de conocimiento declarativo podría ser la capacidad de recordar el nombre de las personas, números de teléfono, direcciones, entre otros.

   En el campo de la Inteligencia Artificial, la representación y utilización del conocimiento declarativo son esenciales para que los sistemas inteligentes puedan acceder a la información relevante y comprender el entorno en el que operan. Mediante la utilización de patrones y relaciones almacenados, los algoritmos de IA pueden identificar características significativas, realizar inferencias y resolver

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

problemas específicos. Esta forma de conocimiento es fundamental para que los agentes inteligentes tomen decisiones adecuadas y actúen de manera informada en un amplio rango de situaciones (Baral, 2003).

2) **Conocimiento procedimental:** Este conocimiento implica la habilidad de realizar procesos o acciones específicas para alcanzar objetivos concretos. Se centra en el "saber hacer", esencial para ejecutar tareas y resolver problemas eficientemente.

   Un ejemplo cotidiano sería el manejo de operaciones matemáticas básicas, como sumar, restar, multiplicar o dividir, donde seguimos pasos concretos para llegar a una solución. Similarmente, aprender a conducir implica dominar una serie de procedimientos y técnicas para manejar un vehículo con seguridad.

   En el ámbito de la Inteligencia Artificial (IA), el conocimiento procedimental es vital para desarrollar sistemas avanzados capaces de realizar tareas complejas. Los algoritmos de IA adquieren este conocimiento a través del entrenamiento con grandes volúmenes de datos y la observación de patrones y ejemplos. Una vez internalizado, este conocimiento permite a los sistemas de IA aplicar procedimientos aprendidos para solucionar problemas y tomar decisiones en una variedad de contextos, imitando la capacidad de adaptación y aprendizaje de un ser humano experimentado.

   Este enfoque no solo mejora la claridad y precisión del texto, sino que también incorpora aspectos clave de la IA, como el aprendizaje a partir de datos y la capacidad de adaptación, para una comprensión más profunda del conocimiento procedimental en este campo.

3) **Conocimiento heurístico:** Este conocimiento se refiere a las estrategias y reglas empíricas utilizadas por los seres humanos para solucionar problemas complejos y tomar decisiones de manera ágil y efectiva. Las heurísticas, consideradas atajos mentales, se fundamentan en la experiencia y el juicio práctico, permitiendo alcanzar soluciones adecuadas sin un análisis detallado de todas las posibles alternativas (Edelkamp & Schrödl, 2011).

   Un claro ejemplo de conocimiento heurístico es la resolución de rompecabezas o acertijos mediante métodos previamente exitosos en situaciones análogas. En la vida cotidiana, este tipo de conocimiento se manifiesta en decisiones de compra, donde las personas recurren a reglas prácticas, como preferir marcas reconocidas o buscar ofertas, para facilitar sus elecciones.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

En el contexto de la Inteligencia Artificial, las heurísticas son igualmente significativas. Los sistemas de IA pueden incorporar técnicas heurísticas para optimizar la búsqueda de soluciones en escenarios complejos, como en el desarrollo de algoritmos de búsqueda o planificación. En el aprendizaje automático, los modelos de IA a menudo aplican heurísticas para mejorar su rendimiento y eficiencia en el tratamiento de datos.

Sin embargo, es necesario reconocer que el conocimiento heurístico, aunque provechoso en numerosas situaciones, puede conducir a sesgos y errores en la toma de decisiones. Por ello, se recomienda combinar el uso de heurísticas con un enfoque analítico y metódico cuando sea necesario, buscando un equilibrio entre rapidez y precisión en el proceso de decisión.

1.9 Representación del conocimiento

La representación del conocimiento en IA y su relación con la ciencia cognitiva es profunda y multifacética. Los seres humanos procesan y almacenan información de diversas maneras: mediante imágenes mentales, lenguaje (oral o escrito), representaciones gráficas, y más. Este conocimiento puede manifestarse de distintas formas, como cadenas de caracteres, o como señales eléctricas o magnéticas en computadoras.

En el contexto de la IA, el desafío radica en la transformación de este conocimiento, naturalmente comprensible para los humanos, en un formato accesible y utilizable por sistemas de inteligencia artificial. Esto implica codificar el conocimiento de tal manera que permita a la IA reconocerlo, procesarlo y, cuando sea necesario, inferir nuevo conocimiento a partir de él.

La representación del conocimiento en IA típicamente se concreta a través de formatos escritos y estructuras de datos específicas, como bases de datos, árboles de decisión o redes semánticas, que facilitan su almacenamiento en sistemas informáticos y su manipulación mediante software. La elección de la forma de representación y las estructuras de datos depende tanto del tipo de problema a abordar como de las técnicas de inferencia que se utilizarán para resolverlo (Ramirez, 2012).

El propósito principal de esta representación es dotar a entidades inteligentes, tales como programas con bases de conocimientos, de la capacidad para tomar decisiones informadas y adaptativas en su entorno. Esta capacidad no se limita a la mera acumulación

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

de datos: implica comprender y aplicar relaciones, normas y patrones en contextos cambiantes, lo cual es fundamental para el razonamiento, el aprendizaje y la adaptación de los sistemas de IA.

Esta perspectiva ampliada resalta la complejidad de trasladar el conocimiento humano a un formato que sea operativo para la IA, enfatizando la importancia de una representación del conocimiento que sea tanto comprensible como funcional para los sistemas inteligentes.

1.9.1 Métodos de representación del conocimiento

A continuación, se listan algunos métodos que se utilizan para representar el conocimiento en la inteligencia artificial:

a) Las ***redes semánticas*** (figura 1.15) se presentan como una herramienta valiosa para representar las interconexiones entre varios objetos, conectando conceptos o nodos a través de relaciones, lo que permite representar información de manera organizada y estructurada (Markman, 2013). Este enfoque guarda semejanza con una característica fundamental de la memoria humana, en la que se tejen numerosas relaciones, de modo que pensar en un concepto puede evocar una multitud de otros. Por ejemplo, si en la red semántica se sabe que "Una ballena es un mamífero" y "Un mamífero es un animal", se puede inferir que "Una ballena es un animal". La verdadera fortaleza de estas redes radica en su capacidad para facilitar la inferencia a partir del conocimiento representado.

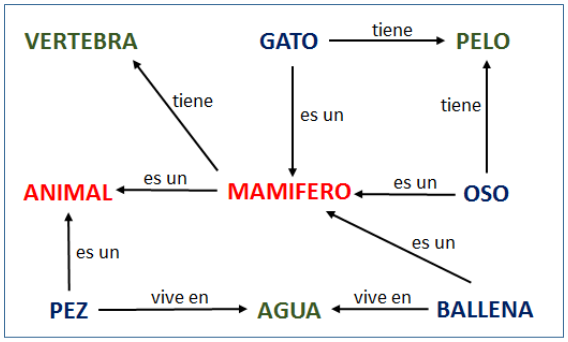


Figura 1.15 Red semántica

Elaborado por los Autores

Las redes semánticas facilitan la interpretación del significado en textos al visualizar las conexiones entre conceptos distintos. Un ejemplo sería representar la frase 'Los gatos cazan ratones' mediante nodos correspondientes a 'gatos', 'cazan' y 'ratones', delineando claramente las relaciones entre estos elementos."

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Las redes semánticas se emplean en bases de datos y la web semánticas para representar y consultar información (Weller, 2010). Sirven como entrada para algoritmos de aprendizaje automático, facilitando que las máquinas entiendan relaciones entre conceptos. Se puede también utilizar para comprender la relación entre diferentes productos o contenidos, por ejemplo, pueden las redes relacionar películas según género, actores y directores. Además, los robots pueden usarlas para entender el mundo que los rodea, reconociendo objetos y comprendiendo las relaciones entre ellos.

b) Los ***marcos*** se especializan en describir las propiedades y relaciones de un concepto específico. Piense en los marcos como "fichas de información" de un objeto o concepto específico. Un marco es esencialmente una estructura de datos que representa un concepto estereotipado. Por ejemplo, un marco para "automóvil" podría contener atributos como "año", "color", "modelo" y "fabricante", y especificar relaciones como "Propietario: quien posee el automóvil" o "clases de automóvil: sedan, suv, deportivo, hatchback, crossover, familiar".

Si tomamos como ejemplo el concepto de "edificio", un marco relacionado con este término podría incluir aspectos específicos como "cantidad de niveles", "material predominante" y "fecha de construcción", y podría formar vínculos como "ubicado en" para señalar una localización o "diseñado por" para mencionar al arquitecto responsable. En contraste, una red semántica enfatizaría las relaciones del concepto "edificio" con otros términos vinculados, tales como "ciudad", "estilo arquitectónico" o "uso residencial". Es decir, mientras el marco se enfoca en profundizar en las características particulares de un concepto, la red semántica ilustra las interacciones de este concepto con otros en un contexto más global.

Los sistemas expertos utilizan marcos para organizar el conocimiento de un área determinada. Por ejemplo, en el ámbito médico, el marco "Enfermedad" podría contener atributos como síntomas, causas y tratamientos. Dentro del Procesamiento del Lenguaje Natural, los marcos facilitan la comprensión de enunciados; como en "Xavier compró un automóvil", donde se identifican roles semánticos: comprador (Xavier), objeto (automóvil) y acción (comprar). En el diseño de videojuegos, un marco podría caracterizar entidades como "Personaje", definiendo aspectos como su contextura o habilidades. En diseño de interfaces, un marco referente a un "Botón"

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

podría especificar su posición, tamaño y función. Finalmente, en robótica se puede usar para reconocer y actuar en diferentes esenarios, ejemplo un marco "Obstáculo" ofrecería detalles sobre su forma, tamaño y estrategias para esquivarlo.

c) La ***lógica proposicional*** es la forma más básica de lógica, donde cada proposición tiene un valor de verdad. Se utiliza principalmente para hacer inferencias. Una proposición es una declaración que puede ser cierta o incorrecta, pero no puede ser ambas cosas simultáneamente. Por ejemplo, "Hoy es sábado" es una proposición porque, dependiendo del día, podría ser verdadera o falsa.

   La lógica proposicional es esencial en el diseño y análisis de circuitos lógicos en electrónica, en dominios donde las decisiones se pueden tomar basadas en reglas y hechos concretos, ejemplo, un sistema que diagnostica enfermedades en base a síntomas: "Si tiene fiebre y tos, entonces puede tener gripe". También se aplica en la resolución de problemas y juegos de estrategia, como el tres en línea, donde se evalúan situaciones con valores de verdad. Es visible en estructuras de control como “if-else", donde, por ejemplo, se plantea que "si el usuario se autentica, entonces puede acceder al sistema". Además, es fundamental en juegos que demandan estrategias y un razonamiento lógico sustentado en normas explícitas.

d) La ***lógica de predicados*** representa el conocimiento a través de proposiciones y cuantificadores. Esto permite realizar inferencias lógicas y deducciones formales. Por ejemplo, a partir de las proposiciones "Todos los humanos son mortales" y "Leonardo da Vinci es humano", se puede inferir que " Leonardo da Vinci es mortal".

   En el ámbito del procesamiento del lenguaje natural, la lógica de predicados se aplica a la interpretación semántica de las oraciones. Un claro ejemplo es la oración "María quiere a Patricio" que puede representarse como Quiere(María, Patricio). Las consultas en bases de datos, especialmente con lenguajes como SQL, a menudo se basan en principios de lógica de predicados. un ejemplo sería buscar todos los libros de un autor determinado. Esta lógica también es esencial en la planificación automática y la solución de problemas, debido a que permite razonar sobre acciones, sus precondiciones y resultados. Lenguajes de programación como Prolog están estructurados en base a la lógica de predicados y son ideales para abordar desafíos que demandan razonamiento lógico y exploración (Fraser, 1990).

   Además, es utilizada en la representación de conocimiento en ontologías, donde

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

se esbozan conceptos y sus interconexiones en un dominio concreto. En el campo de la robótica, esta lógica facilita la toma de decisiones y planificación, al razonar sobre el estado del entorno, las acciones viables y sus implicaciones.

e) Las ***Reglas de Producción***, son una forma de representar el conocimiento en sistemas expertos y otras áreas de la inteligencia artificial. Estas reglas tienen una estructura condicional, y se pueden entender como instrucciones que indican qué hacer o qué inferir bajo ciertas condiciones. Las reglas tienen el formato "SI ... ENTONCES ...". Por ejemplo, una regla podría ser: "SI es mamífero ENTONCES es un animal". Las reglas de producción hacen explícito el conocimiento. Esto permite entender fácilmente cómo se toma una decisión o se llega a una conclusión.

   En la robótica y en el control de procesos industriales, las reglas de producción son esenciales para establecer respuestas adaptativas ante condiciones concretas del entorno. Al reconocer ciertos patrones, es posible activar reglas que categorizan o determinan decisiones basadas en esos patrones identificados. En el mundo de los videojuegos, estas reglas modelan las acciones de personajes no jugables y establecen dinámicas del juego. En el ámbito de las simulaciones, las reglas de producción orientan las acciones de los agentes dentro de un sistema virtual. Mientras que en el e-learning, estas reglas guían el proceso educativo, ofreciendo retroalimentación y sugerencias al estudiante.

f) Las ***Redes Bayesianas*** son estructuras gráficas en forma de grafos acíclicos dirigidos que delinean conjuntos de variables junto con sus interdependencias condicionales, derivadas de una factorización de la función de distribución conjunta. Estas redes permiten representar y procesar información bajo condiciones de incertidumbre (Croitoru et al., 2018). Proporcionan una forma estructurada para analizar cómo diferentes variables se relacionan entre sí y cómo la evidencia o información sobre una variable puede modificar nuestras percepciones sobre otras variables.

   Considere tres variables: "Nublado", "Lluvia" y "Césped mojado". En este escenario, la probabilidad de que el "Césped esté mojado" está influenciada por la "Lluvia". Sin embargo, "Nublado" puede condicionar esta relación, debido a que, si el día está nublado, la probabilidad de lluvia aumenta, y consecuentemente, la del césped mojado.

   En el sector médico, estas redes facilitan la estimación de la probabilidad de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

padecer una enfermedad considerando diversos síntomas presentes. En el campo de la genética, permiten descifrar las interacciones entre genes y determinar cómo éstos pueden estar relacionados con determinadas enfermedades. Estas redes son esenciales en la detección de anomalías y en sistemas avanzados de reconocimiento facial. Gracias a su capacidad de análisis, pueden ofrecer recomendaciones personalizadas de productos o contenidos a los usuarios, tomando en cuenta sus historiales y preferencias. Además, son herramientas valiosas para prever y diagnosticar fallos en sistemas complejos, tales como redes eléctricas o maquinarias industriales. En robótica, apoyan en la percepción sensorial y en la toma de decisiones cuando los robots se enfrentan a situaciones inciertas. Finalmente, son fundamentales en la predicción meteorológica y en la simulación de la propagación de enfermedades contagiosas.

g) Los ***Modelos de Markov***, son modelos estocásticos matemáticos que representan sistemas que evolucionan a lo largo del tiempo donde la probabilidad de transición de un estado a otro solo depende del estado actual y no de cómo llegó a ese estado, por lo que se utilizan para predecir estados futuros basados en estados actuales, sin tener en cuenta su historia pasada. Su característica principal es la "falta de memoria".

Cuando se lanza una moneda, tiene una probabilidad del 50% de caer en cara y una probabilidad del 50% de caer en cruz. Si se decide jugar un juego en el que se lanza la moneda repetidamente y se registra la secuencia de resultados. Al usar un Modelo de Markov para predecir el resultado del próximo lanzamiento, solo se basaría en el resultado del último lanzamiento. No importaría si, en los últimos 12 lanzamientos, se obtuvo 9 caras y 3 cruz. Suponga que el último lanzamiento fue cara. Según el modelo, la probabilidad de que el próximo lanzamiento sea una cara sigue siendo del 50%, y la probabilidad de que sea cruz también es del 50%. El modelo no "recuerda" los lanzamientos anteriores. En situaciones reales, los Modelos de Markov pueden abordar sistemas con múltiples estados y con probabilidades de transición entre ellos mucho más elaboradas.

En el ámbito del procesamiento de lenguaje natural, los modelos de Markov son empleados para construir y perfeccionar modelos de n-gramas, que son esenciales para prever la siguiente palabra en una secuencia textual, basándose en la probabilidad de aparición de ciertas palabras consecutivas. Estos modelos son igualmente valiosos en

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

bioinformática, donde se utilizan para la predicción de secuencias de ADN o proteínas, así como para simular las transiciones entre distintos estados conformacionales de una molécula. En el contexto clínico, los modelos de Markov ofrecen una estructura para diseñar modelos de decisión en tratamientos médicos, permitiendo representar las transiciones entre los distintos estados de salud de un paciente y las probabilidades asociadas a cada cambio.

En el terreno de las telecomunicaciones, estos modelos facilitan la modelización del tráfico de redes, pronosticando la dinámica de la red bajo variadas condiciones. Son también fundamentales en la ciencia de datos aplicada a juegos estratégicos como el ajedrez o el póker, donde se analizan y desarrollan tácticas basadas en la probabilidad. La confiabilidad y el mantenimiento de sistemas mecánicos y equipos también se benefician de la aplicación de modelos de Markov para predecir fallos y optimizar programas de mantenimiento. Finalmente, en la robótica, contribuyen a la planificación y optimización de rutas para robots autónomos, adaptándose a nuevos entornos y desafíos (Eze, 2022).

h) Los ***Árboles de Decisión***, son herramientas de soporte a la decisión que utilizan un modelo de decisiones y sus posibles consecuencias. Poseen una estructura gráfica que emula la toma de decisiones humanas, permitiendo visualizar de manera jerárquica múltiples opciones y sus posibles resultados. En el campo del aprendizaje automático, los árboles de decisión son un tipo de modelo utilizado para clasificación y regresión. Un ejemplo (figura 1.16) pude ser decidir si jugar futbol o no en base a las variables llover y viento.

   El nodo raiz podría plantear la pregunta ¿Está lloviendo? Si la respuesta es afirmativa, podrías optar por no jugar al fútbol, pero si es negativa, avanzarías al siguiente nodo: "¿Hay viento?". En caso de que la respuesta sea afirmativa, podrías decidir no jugar debido a que el viento podría influir en el juego; si es negativa, entonces elegirías jugar al fútbol.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Figura 1.16 Árbol de decisión

Elaborado por los Autores

Las entidades financieras se valen de árboles de decisión para evaluar la viabilidad de conceder créditos, integrando variables como el nivel de ingresos, el historial crediticio y la edad del solicitante. Esta herramienta analítica también se aplica para la segmentación de clientes y la personalización de estrategias publicitarias; por ejemplo, a un cliente que previamente ha adquirido indumentaria deportiva se le podrían mostrar anuncios de calzado deportivo, tras verificar sus compras anteriores.

En el procesamiento de lenguaje natural, los árboles de decisión facilitan la interpretación de palabras pronunciadas por un usuario, considerando elementos acústicos y el contexto lingüístico. En el ámbito agrícola, contribuyen a tomar decisiones sobre la selección de cultivos, ponderando factores como las características del suelo, la estacionalidad y la susceptibilidad a enfermedades. Las tiendas online implementan árboles de decisión para sugerir productos, basándose en el análisis del comportamiento de navegación y compras previas de los usuarios. Tambien, en tareas de clasificación y regresión, estos modelos pueden discernir, por ejemplo, si un correo electrónico es spam, evaluando sus atributos distintivos.

i) Las ***Ontologías***, se refieren a una representación formal y estructurada de un conjunto de conceptos dentro de un dominio y las relaciones entre esos conceptos. Son utilizadas en informática y en la web semántica para estructurar y clasificar información. La web semántica es una extensión de la web actual que busca hacer que los datos sean comprensibles no solo para humanos, sino también para máquinas. Las ontologías proporcionan un marco estructurado que permite a las máquinas entender y razonar sobre los datos.

La web semántica se apoya en ontologías para dotar de significado a los datos de Internet, facilitando así que las máquinas procesen y respondan ante consultas de alta complejidad. Un caso representativo es el proyecto DBpedia, que convierte la información de Wikipedia en datos estructurados accesibles para interrogaciones

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

semánticas. En el comercio electrónico, las ontologías son utilizadas para la organización y categorización de productos, posibilitando recomendaciones más acertadas y una búsqueda más eficiente para los consumidores. Además, estas estructuras son herramientas poderosas para la organización, clasificación y recuperación de información; por ejemplo, pueden vincular términos de distintos idiomas o establecer conexiones entre autores, sus obras y temáticas correspondientes.

En el ámbito empresarial, las ontologías se emplean para la modelización de procesos de negocio, recursos y servicios, contribuyendo a optimizar la gestión y el proceso de toma de decisiones. Son también aliadas en la integración y análisis de datos de diversas fuentes, lo cual es decisivo para el estudio de fenómenos complejos como el cambio climático. Las ontologías constituyen un pilar en el desarrollo de sistemas expertos y asistentes virtuales, al permitir que las máquinas interpreten y razonen sobre dominios específicos. En el sector turístico, se utilizan para modelar y relacionar datos sobre destinos, actividades y alojamientos, entre otros, brindando así recomendaciones personalizadas a los viajeros.

j) Las ***Redes Neuronales Artificiales***, son modelos computacionales inspirados en cómo funcionan las redes neuronales biológicas del cerebro humano. Se pueden ver como un medio de representar el conocimiento a través de conexiones ponderadas entre neuronas que están organizadas. Una neurona recibe uno o más inputs, los procesa y produce un output. Cada conexión entre neuronas tiene un peso asociado, que puede ser positivo (excitatorio) o negativo (inhibitorio). Estos pesos se ajustan durante el proceso de aprendizaje. La distribución del conocimiento a través de la red permite a las redes neuronales artificiales generalizar y hacer predicciones sobre datos nunca antes vistos.

Las Redes Neuronales Artificiales (RNA), y en particular las Redes Neuronales Convolucionales (CNN), son fundamentales en la identificación de objetos dentro de imágenes, el reconocimiento facial y el diagnóstico médico mediante el análisis de imágenes clínicas. Por otro lado, las Redes Neuronales Recurrentes (RNN), junto con los modelos de atención y las arquitecturas Transformer, se destacan en aplicaciones de procesamiento del lenguaje natural, tales como la traducción automática, la creación de contenido textual y el análisis de sentimiento. Además, las RNA son trascendentales en el desarrollo de asistentes virtuales y en la transcripción automática de señales de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

audio.

En la robótica, son pieza clave para el procesamiento de datos sensoriales y la toma de decisiones en vehículos autónomos (Eze, 2022). En el sector de la salud, las RNA permiten el reconocimiento y diagnóstico preciso de enfermedades a través de imágenes médicas, el análisis de secuencias genéticas y la identificación de patrones en registros de salud electrónicos. Específicamente, las Redes Generativas Adversarias (GAN) tienen la notable capacidad de generar imágenes, música y textos indistinguibles de aquellos creados por seres humanos, abriendo así nuevas fronteras en la creación de contenido digital.

k) Los ***Marcos Temporales***, son sistemas de representación de conocimiento que se enfocan específicamente en capturar y manejar la información relacionada con el tiempo, como eventos que ocurren antes, después o durante otros eventos. La temporalidad es una dimensión importante en muchos sistemas y aplicaciones, y entender las relaciones temporales entre eventos puede ser vital para decisiones, razonamientos y predicciones. Se aplican en la elaboración de planes, en los cuales es esencial saber qué tarea precede o sigue a otra para asignar recursos o tiempo de manera efectiva.

En el ámbito médico, la comprensión detallada de la cronología de los síntomas y los tratamientos es esencial para el diagnóstico y la terapia adecuados, lo que subraya la importancia del análisis de series temporales. Estas series son igualmente críticas en la organización y secuenciación de tareas según la temporalidad, como ocurre en la logística del transporte o en la programación de la producción industrial. En el terreno de los videojuegos y las simulaciones, los agentes informáticos emplean marcos temporales para determinar acciones, basándose en la interpretación de eventos anteriores, actuales y proyectados. Las bases de datos especializadas en series temporales son importantes en la gestión de datos dinámicos, como en el seguimiento de transacciones financieras o en el mantenimiento de registros médicos. Este tipo de análisis temporal es también fundamental en sistemas de reservación, como los de hoteles y vuelos, que gestionan la disponibilidad y las reservas en distintos períodos.

En la industria, se utiliza para monitorear y evaluar el flujo de datos de sensores y actuadores a lo largo del tiempo, proporcionando así información valiosa para el mantenimiento y la optimización de procesos. Además, es instrumental para el

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

seguimiento y análisis de variables ambientales, permitiendo una mejor respuesta frente a fenómenos como el cambio climático.

Los marcos temporales facilitan la estructuración y secuencia de eventos en narrativas, ya sea en literatura, cine o videojuegos, aportando coherencia y continuidad a las historias. En situaciones críticas como la gestión de crisis o emergencias, el análisis temporal es vital para tomar decisiones basadas en la evolución de los eventos. Asimismo, en la logística, las series temporales son indispensables para rastrear la distribución y entrega de paquetes, asegurando eficiencia y efectividad en la cadena de suministro.

l) La ***Representación Basada en Casos***, en lugar de utilizar reglas generales, esta representación utiliza casos específicos o ejemplos previos para tomar decisiones, se resuelve nuevos problemas basándose en soluciones a problemas anteriores. Si un problema fue resuelto en el pasado, una situación similar en el presente puede ser abordada de manera similar, con algunas adaptaciones si es necesario. Si la solución es exitosa, el nuevo problema y su solución se almacenan como un nuevo caso en la base de casos.

El razonamiento basado en casos (RBC) es una herramienta poderosa en la medicina, debido a que permite a los doctores acceder a un repositorio de casos clínicos previos, analizando síntomas y diagnósticos similares para optimizar el proceso de diagnóstico actual. En el ámbito corporativo, las empresas pueden implementar sistemas de RBC para mejorar el servicio al cliente, aplicando soluciones efectivas de situaciones pasadas a consultas nuevas.

Los ingenieros, al enfrentarse a desafíos de diseño, pueden utilizar bases de datos de casos anteriores para inspirarse y encontrar soluciones innovadoras a problemas actuales. El comercio electrónico se beneficia enormemente del RBC, personalizando recomendaciones de productos para clientes basándose en el historial de compras y búsquedas de usuarios con patrones similares.

Los urbanistas y arquitectos tienen la posibilidad de consultar proyectos previos para informar y enriquecer el diseño y planificación de nuevos espacios urbanos. En el contexto educativo, el RBC puede facilitar la personalización del aprendizaje, ofreciendo recursos y ejercicios ajustados a las necesidades individuales, identificadas a través de desafíos enfrentados por estudiantes en temas equivalentes. En el ámbito

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

legal, abogados y jueces pueden aprovechar el análisis de precedentes legales mediante RBC para fundamentar decisiones judiciales o desarrollar argumentos robustos en casos actuales. Las entidades financieras pueden mejorar la precisión en la evaluación de solicitudes de crédito al comparar con historiales financieros análogos. Finalmente, en la agricultura, el RBC puede ser dignificativo para identificar rápidamente y tratar enfermedades o plagas, aplicando estrategias de manejo probadas en situaciones similares previas, lo que resulta en una gestión agrícola más eficiente y efectiva.

Debido a la diversidad de formas que puede tomar el conocimiento, los desafíos inherentes al desarrollo de una representación del conocimiento son complejos, interrelacionados y dependen de los objetivos específicos. En términos generales, el objetivo es lograr que el conocimiento esté representado de tal manera que:

1. Capturar generalizaciones significativas que faciliten la extrapolación y aplicación en distintos contextos, manteniendo la pertinencia y la utilidad a pesar de la variabilidad de las situaciones.
2. Ser accesible y comprensible tanto para los expertos que lo proporcionan como para los sistemas que lo procesan, promoviendo una sinergia entre el conocimiento humano experto y el procesamiento algorítmico.
3. Permitir modificaciones y actualizaciones de manera ágil y sin fricciones, adaptándose a la evolución del conocimiento y a las nuevas informaciones que surgen continuamente.
4. Ser aplicable en una amplia gama de situaciones, incluso en casos donde la información no es completa o exacta, demostrando flexibilidad y robustez en la inferencia bajo incertidumbre.
5. Servir como un filtro eficaz para acotar las alternativas durante la búsqueda de soluciones, optimizando así los procesos de toma de decisiones y resolución de problemas.

Estos principios buscan que la representación del conocimiento en IA sea una herramienta dinámica y adaptativa, capaz de soportar procesos complejos de razonamiento y aprendizaje automático.

En el extenso campo de la inteligencia artificial, la representación del conocimiento es un pilar fundamental que adopta diversas formas para abordar distintos tipos de información y procesamiento:

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

El *conocimiento declarativo* puede representarse mediante *modelos relacionales* en forma de árboles, grafos o redes semánticas y *esquemas basados en lógica* que incluyen la lógica proposicional y lógica de predicados.

El *conocimiento procedimental* puede ser representado para su almacenamiento mediante gramáticas formales implementadas a través de sistemas o lenguajes procedimentales, sistemas basados en reglas (sistemas de producción), diagramas de flujo, máquinas de estado, script.

El *conocimiento heurístico* puede ser representado mediante sistemas basados en reglas, árboles de decisión, redes neuronales, algoritmos genéticos, sistemas expertos, funciones heurísticas, agentes basados en conocimiento, mapas cognitivos (Edelkamp & Schrödl, 2011).

Las representaciones prácticas incluyen elementos tanto declarativos, procedimentales y en ocasiones heurísticos.

1.10 Agente basado en conocimiento

Los agentes basados en conocimiento tienen la capacidad de utilizar el conocimiento expresado en formas altamente genéricas, combinando y recombinando la información para adaptarse a diferentes objetivos. Pueden fusionar el conocimiento general con las percepciones en tiempo real para inferir aspectos ocultos del estado del mundo antes de tomar una decisión sobre cualquier acción.

Se puede establecer una analogía entre un agente basado en conocimiento y un médico que realiza un diagnóstico en un paciente, deduciendo una enfermedad que no es directamente evidente antes de decidir un curso de tratamiento. El médico utiliza su conocimiento, que está representado en forma de reglas adquiridas a partir de libros y la enseñanza de sus profesores, así como patrones de asociación que no puede describir explícitamente.

Los agentes basados en conocimiento son flexibles y pueden asumir nuevas tareas en forma de objetivos explícitamente definidos. Tienen la capacidad de adquirir competencias de manera eficiente al acceder y aprender rápidamente del conocimiento disponible en su entorno. Además, son capaces de adaptarse a cambios en su entorno al actualizar el conocimiento relevante de manera oportuna.

El componente principal de un agente basado en conocimiento es su *base de conocimiento o KB(Knowledge Based)*, que es un conjunto de *sentencias* que representan

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

alguna aserción acerca del mundo. Cada sentencia se expresa en un lenguaje llamado lenguaje de representación del conocimiento. Cuando la sentencia se considera como un hecho sin depender de otras sentencias, se le denomina axioma.

El mecanismo para añadir sentencias nuevas a la base de conocimiento y preguntar qué se sabe en la base de conocimiento es la *inferencia* que permite derivar nuevas sentencias de las antiguas. En los *agentes lógicos* basados en conocimiento la inferencia debe cumplir con el requisito esencial de que cuando se *pregunta* a la base de conocimiento, la *respuesta* debe ser producto del contenido existente en la base de conocimiento. El proceso de inferencia no puede inventar cosas a medida que avanzan las deducciones que realiza.

El siguiente ejemplo ilustra este concepto. Considerar las siguientes sentencias:

1. Si no llovió, Patricio visitó hoy a Victoria.
2. Patricio visitó hoy a Victoria o Benjamín, pero no a ambos.
3. Patricio visitó hoy a Benjamín.

Estas tres sentencias constituyen la descripción en el nivel de conocimiento, donde solo se ha especificado lo que el agente conoce y en función de su objetivo se va a determinar su comportamiento, que en este caso es dar respuesta a la pregunta "¿hoy llovió?", aunque ninguna de las trese esentencias indica si hoy llovió. Así es como se puede hacer: en base a la oración 3 si se sabe que es verdadera, se conoce que Patricio visitó a Benjamín. Mirando la oración 2, se sabe que Patricio visitó a Victoria o Benjamín, y por lo tanto podemos concluir que:

4. Patricio no visitó a Victoria.

Ahora, mirando la oración 1, se entiende que, si no hubiera llovido, Patricio habría visitado a Victoria. Sin embargo, conociendo la oración 4, se sabe que este no es el caso. Por lo tanto, podemos concluir

5. hoy llovió.

Para llegar a esta conclusión, se usa la lógica que se basa en la información existente. Se han usado sentencias que son afirmaciones sobre el mundo en un lenguaje de representación del conocimiento, que permite a la IA almacenar el conocimiento y lo usa para inferir nueva información.

Un agente basado en conocimiento se puede desarrollar de manera efectiva al proporcionarle el conocimiento necesario. Iniciando con una base de conocimiento vacía,

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

el diseñador del agente puede introducir sentencias una a una, permitiendo al agente adquirir gradualmente la capacidad de operar en su entorno. Este enfoque se conoce como el método declarativo para construir el sistema. En contraposición, el enfoque procedimental implica la codificación directa de comportamientos deseados como código de programa.

Un agente exitoso a menudo combina elementos tanto declarativos como procedimentales en su diseño, y el conocimiento declarativo a menudo se puede compilar en un código procedimental más eficiente.

Asimismo, es posible dotar a un agente basado en conocimiento de mecanismos que le permita aprender de manera autónoma, generando conocimiento general sobre su entorno a partir de una secuencia de percepciones. Un agente de aprendizaje puede ser totalmente autónomo.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

CAPITULO 2

2 FUNDAMENTOS DE LÓGICA Y REGLAS DE PRODUCCIÓN

La lógica y las reglas de producción son los pilares fundamentales en el campo de la inteligencia artificial, que proporcionan las bases lógicas y algorítmicas que permiten a las máquinas razonar, tomar decisiones y resolv(Feigenbaum & others, 1977)iente y eficaz (Feigenbaum & others, 1977). Se exploran a fondo, los fundamentos de la lógica que constituye el lenguaje formal que se emplea para expresar declaraciones y razonar sobre su verdad o falsedad, y las reglas de producción que permiten automatizar tareas complejas. Comprender estos conceptos es importante para diseñar sistemas de IA y para resolver problemas del mundo real.

Se parte de la lógica proposicional que se centra en la relación entre proposiciones simples y cómo se combinan para formar proposiciones más complejas mediante operadores lógicos. Luego se avanza a la lógica de primer orden, que permite la representación de estructuras más complejas y el uso de variables y cuantificadores para expresar relaciones entre objetos (Brachman & Levesque, 2004a).

También se aborda el concepto de cuantificadores, que permiten cuantificar conjuntos de elementos y expresar afirmaciones generales. Esto es esencial para el razonamiento deductivo, que implica llegar a conclusiones lógicas a partir de premisas dadas. Se examina las reglas de producción, un enfoque práctico en la IA que se basa en el uso de reglas condicionales para tomar decisiones y realizar acciones. Estas reglas son esenciales para la automatización de tareas y la toma de decisiones en sistemas inteligentes.

2.1 Lógica

La lógica, como disciplina, tiene sus raíces en Aristóteles, pero fue Leibniz quien la desarrolló desde una perspectiva matemática y la denominó "lógica matemática". Esta disciplina es esencial en el ámbito de la inteligencia artificial, y su aplicación se extiende ampliamente a diversas áreas, como semasiología, programación lógica, teoría de especificación de software y validación, bases de datos, representación del conocimiento, sistemas inteligentes y robótica. La lógica es fundamental en la construcción y desarrollo de sistemas inteligentes y en la comprensión de la inteligencia artificial en general.

La lógica en la inteligencia artificial requiere de varias partes:

La **sintaxis** es importante en la determinación de la corrección formal de todas las

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

sentencias en un lenguaje, y esta idea es especialmente relevante en el contexto de los lenguajes de programación (Körner et al., 2022). Las bases de conocimiento están compuestas por sentencias que se expresan de acuerdo con la sintaxis del lenguaje de representación que se utiliza. Como ejemplo, para representar la afirmación "Juan es el padre de Alfonso" en un lenguaje como Prolog, se utilizaría la siguiente sintaxis: Padre (Juan, Alfonso).

La ***semántica*** trata tarta sobre el significado de las sentencias y su valor de verdad con relación al modelo del mundo en el que existe. Este valor de verdad es de naturaleza binaria, es decir, una sentencia puede ser verdadera o falsa en un modelo de mundo específico.

En el contexto de la representación del conocimiento, como en el ejemplo dado Padre (Juan, Alfonso), es importante entender que la verdad de una afirmación puede variar según el modelo del mundo o el contexto en el que se evalúe. En el modelo del mundo posible diseñado para un sistema de representación de conocimiento, Padre (Juan, Alfonso) puede considerarse verdadero, incluso si no es cierto en el mundo real. Esto resalta la idea de que el valor de verdad de una afirmación puede ser diferente en diferentes contextos o modelos de mundo, lo que es fundamental en la lógica y la representación del conocimiento.

El **modelo** del mundo es una abstracción matemática que se utiliza para definir la verdad o falsedad de cada sentencia en el contexto de la representación del conocimiento. Este modelo se expresa mediante variables que pueden tomar diferentes valores, lo que permite representar una variedad de situaciones y relaciones.

En el ejemplo que estamos analizando, la sentencia Padre (x, y) se utiliza para expresar la relación padre-hijo entre dos variables, x e y. Estas variables pueden tomar diferentes valores, como "Juan" y "Alfonso" en este caso se tendría que Padre (Juan, Alfonso), o cualquier otro par de valores, como en "Alfonso" y "Ximena" que sería expresado como Padre(Alfonso, Ximena).

Este enfoque basado en variables y modelos de mundo permite representar y razonar sobre una amplia gama de situaciones y relaciones en la inteligencia artificial y la representación del conocimiento.

El razonamiento lógico en la representación del conocimiento se basa en la relación de **implicación** lógica entre sentencias. En notación matemática, esta relación se

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

denota como $\alpha \rightarrow \beta$, lo que significa que, si la sentencia α es verdadera, entonces la sentencia β también debe ser verdadera. En el contexto de una base de conocimiento que contiene hechos verdaderos que representan un modelo del mundo, se pueden deducir nuevas afirmaciones lógicas utilizando reglas y relaciones lógicas.

En el ejemplo, si la base de conocimiento contiene las sentencias Padre (Juan, Alfonso) y Padre(Alfonso, Ximena), se puede deducir la relación de abuelo siguiendo el concepto de que un abuelo es el padre del padre de una persona. Por lo tanto, se puede concluir que Juan es el abuelo de Ximena, debido a que Alfonso es el padre de Ximena, y Juan es el padre de Alfonso. Este tipo de razonamiento lógico es fundamental en la representación del conocimiento y la inteligencia artificial para inferir nuevas relaciones y conocimientos a partir de la información existente en una base de conocimiento.

La **solidez** del algoritmo de inferencia es fundamental para garantizar que las implicaciones lógicas siempre sean verdaderas y coherentes (García Serrano, 2012). En el ejemplo que se menciona, la inferencia de la relación de "abuelo" se basa en un concepto sólido y bien definido: que un abuelo es el padre del padre de una persona. Si el algoritmo de inferencia está diseñado correctamente y sigue reglas lógicas sólidas, entonces esta deducción será siempre válida y producirá resultados precisos y coherentes.

La robustez y la solidez de los algoritmos de razonamiento lógico son aspectos clave en la representación del conocimiento y la inteligencia artificial, debido a que permiten que los sistemas deduzcan nuevas relaciones y conocimientos de manera confiable a partir de la información disponible en la base de conocimiento. Esto es esencial para tomar decisiones informadas y realizar tareas de razonamiento de manera efectiva.

La **completitud** es una propiedad importante en un algoritmo de inferencia. Un algoritmo de inferencia completo garantiza que todas las sentencias que están implicadas lógicamente en la base de conocimiento se derivarán correctamente. En el ejemplo, si se tiene las sentencias Padre (Juan, Alfonso) y Padre(Alfonso, Ximena) en la base de conocimiento y se quiere inferir la relación de "abuelo", un algoritmo completo debería ser capaz de derivar que Juan es el abuelo de Ximena, debido a que esto está implicado lógicamente por las sentencias dadas.

La completitud asegura que no se omita ninguna inferencia válida, lo que es esencial para que un sistema de representación del conocimiento sea confiable y preciso.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Esto significa que el algoritmo de inferencia considerará todas las posibles derivaciones lógicas y no dejará ninguna sin explorar, asegurando que las conclusiones sean exhaustivas y coherentes.

La **denotación** es esencial para establecer la conexión entre los procesos de razonamiento lógico y el entorno real en el que se encuentra un agente o sistema. En el ejemplo que se menciona, la afirmación de que "Juan es el padre de Alfonso" y "Alfonso es el padre de Ximena" tiene una denotación en la vida real, lo que significa que existe una familia en la que estas afirmaciones son verdaderas y reflejan relaciones familiares reales.

La denotación asegura que las inferencias y conclusiones que se hacen en el contexto del razonamiento lógico sean aplicables y relevantes en el mundo real. Esto garantiza que los sistemas de inteligencia artificial puedan tomar decisiones y realizar acciones basadas en un conocimiento que refleja de manera precisa el entorno en el que operan.

2.1.1 Lógica proposicional

La lógica proposicional es un sistema formal en el que el conocimiento se representa mediante proposiciones, que son afirmaciones o declaraciones que pueden ser verdaderas o falsas. Estas proposiciones pueden combinarse de diversas maneras utilizando operadores lógicos como "y" (conjunción), "o" (disyunción) y "no" (negación).

Las expresiones resultantes en la lógica proposicional pueden interpretarse como reglas de inferencia que preservan la verdad, lo que significa que se pueden utilizar para deducir nuevas afirmaciones verdaderas a partir del conocimiento existente o para evaluar la validez de las afirmaciones existentes.

Este enfoque formal es importante en la representación del conocimiento y el razonamiento en la inteligencia artificial, donde se utilizan lógicas proposicionales para modelar y resolver problemas de manera lógica y coherente.

Una proposición es una declaración o una oración declarativa simple que puede ser evaluada como verdadera o falsa. Por ejemplo, la proposición "el carro es lujoso" puede ser verdadera o falsa según las características del carro en cuestión y cómo se defina "lujoso". Además, una proposición puede negarse, lo que daría como resultado la proposición opuesta, en este caso, "el carro no es lujoso".

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Las proposiciones en la lógica proposicional se utilizan para representar afirmaciones y hechos que son la base del razonamiento lógico en la inteligencia artificial y otros campos relacionados.

Tabla 2-1 Conectores básicos de la lógica proposicional

Nombre	*Conector*	*Símbolo*
Disyunción	*OR*	∨
Conjunción	*AND*	∧
Negación	*NOT*	¬
Implicación	*IF – THEN*	→
Bicondicional	*IF-Then en ambas direcciones*	↔

Elaborado por los Autores

Las proposiciones pueden combinarse para formar proposiciones compuestas, lo que permite un razonamiento más complejo. Esto se logra mediante el uso de conectivos proposicionales (Tabla 2-1) que permiten combinar, negar o relacionar proposiciones simples para crear proposiciones más elaboradas. Los conectivos básicos de la lógica proposicional son fundamentales en este proceso y se utilizan para construir expresiones lógicas.

Las tablas de verdad (Tabla 2-2) son herramientas importantes para comprender cómo se comportan las operaciones básicas de la lógica proposicional y cómo afectan la verdad o falsedad de las proposiciones compuestas.

Estas tablas muestran todas las combinaciones posibles de valores de verdad para las proposiciones involucradas y el valor de verdad resultante de la proposición compuesta. Son una herramienta esencial para el análisis lógico y la evaluación de la validez de argumentos lógicos.

Tabla 2-2 Tablas de verdad para operadores lógicos

p	**q**	**Disyunción** $p \vee q$	**Conjunción** $p \wedge q$	**Negación** $\neg p$	**Implicación** $p \rightarrow q$	**Bicondicional** $p \leftrightarrow q$
V	V	V	V	F	V	V
V	F	V	F	F	F	F
F	V	V	F	V	V	F
F	F	F	F	V	V	V

Elaborado por los Autores

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

El primer conectivo de la tabla 2-2 llamado disyunción, es cierto si al menos uno de los disyuntos es verdadero (**p** o **q**), cabe mencionar que existen dos tipos de OR: un OR inclusivo y un OR excluyente. En un OR excluyente, p ∨ q es falso si p ∧ q es verdadero. Es decir, un OR excluyente se requiere que solo uno de sus argumentos sea verdadero y no ambos. Un OR inclusivo es verdadero si cualquiera p, q o p ∧ q es verdadero. Un ejemplo puede ayudar a entender lo inclusivo frente a lo excluyente. Or inclusivo: "para comer el postre, tienes que limpiar tu habitación o lavar los platos". En este caso, si realiza ambas tareas, seguirá recibiendo postre. Or excluyente: "De postre, puedes tomar galletas o helado". En este caso, no puede tener ambos. El OR excluyente a menudo se abrevia como XOR y un símbolo común para él es ⊕.

El segundo es denominado conjunción, es cierto sólo si las dos sentencias son verdaderas (**p** y **q**). Cuando estas dos proposiciones, p y q, están conectadas por ∧, la proposición resultante P ∧ Q es verdadera solo en el caso de que tanto p como q sean verdaderas, en este caso es el AND de los operadores lógicos de la lógica booleana.

El Not (¬) invierte el valor de verdad de la proposición. Así, por ejemplo, si P: "Está lloviendo", entonces ¬P: "No está lloviendo", la tabla muestra la comparación de todas las posibles asignaciones de verdad a las proposiciones. Esta tabla ayuda a comprender mejor los valores de verdad de las proposiciones cuando se conectan con diferentes conectivos lógicos.

El poder de la lógica proposicional entra en juego en el uso de las formas condicionales, cuyas dos formas más básicas se denominan Modus Ponens y Modus Tollens. El conectador de implicación puede ser considerado como un condicional expresado de la siguiente forma, si p→q es verdadero, entonces q debe ser siempre verdadero. Para los casos en los cuales *p* es falso, la expresión *p*→*q*, es siempre verdadera, independientemente de los valores lógicos que tome *q*, debido a que el operador de implicación no puede hacer inferencias acerca de los valores de *q*.

Cuando el antecedente(p) es verdadero, toda la implicación es verdadera en el caso de que el consecuente (q) sea verdadero (eso tiene sentido: si está lloviendo y estoy adentro de la casa, entonces la oración "si está lloviendo, entonces estoy adentro de la casa" es verdadero). Cuando el antecedente (p) es verdadero, la implicación es falsa si el consecuente (q) es falso (si estoy afuera mientras llueve, entonces la oración "Si está lloviendo, entonces estoy adentro" es falsa). Sin embargo, cuando el antecedente (p) es

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

falso, la implicación siempre es verdadera, independientemente del consecuente (q). Esto a veces puede ser un concepto confuso.

Lógicamente, no se puede aprender nada de una implicación (p → q) si el antecedente (p) es falso. Mirando nuestro ejemplo, si no está lloviendo, la implicación no dice nada sobre si estoy adentro o no. Podría ser del tipo de interiores y nunca caminar afuera, incluso cuando no llueve, o podría ser del tipo de exteriores y estar afuera todo el tiempo cuando no llueve. Cuando el antecedente es falso, se dice que la implicación es trivialmente verdadera.

Bicondicional (↔) es una implicación que va en ambas direcciones. Puedes leerlo como "si y solo si". P ↔ Q es lo mismo que P → Q y Q → P juntos. Por ejemplo, si P: "Está lloviendo". y Q: "Estoy adentro", entonces P ↔ Q significa que "Si está lloviendo, entonces estoy adentro", y "si estoy adentro, entonces está lloviendo". Esto significa que podemos inferir más de lo que podríamos con una simple implicación. Si P es falso, entonces Q también es falso; si no llueve, se sabe que tampoco estoy adentro, y esto por tanto es también verdadero.

La lógica proposicional, permite la asignación de un valor verdadero o falso para la sentencia completa, no tiene facilidad para analizar las palabras individuales que componen la sentencia. Por este motivo, la representación de la sentencia del ejemplo "el carro es lujoso" como proposición sería *el_carro_es_lujoso*. Si se combina esta proposición con otra se pueden expresar conceptos más complejos *el_carro_es_lujoso y tiene_reconocimiento_de_voz*. El problema de esta representación es que no es muy expresiva.

El modelo entonces es una asignación de un valor de verdad a cada proposición. Para reiterar, las proposiciones son declaraciones sobre el mundo que pueden ser verdaderas o falsas. Sin embargo, el conocimiento sobre el mundo está representado en los valores de verdad de estas proposiciones. El modelo es la asignación de valores de verdad que proporciona información sobre el mundo.

Por ejemplo, si **P**: "Está lloviendo". y **Q**: "Es miércoles", un modelo podría ser la siguiente asignación de valor de verdad: {P = Verdadero, Q = Falso}. Este modelo significa que está lloviendo, pero no es miércoles. Sin embargo, hay más modelos posibles en esta situación (por ejemplo, {P = Verdadero, Q = Verdadero}, donde está lloviendo un miércoles). De hecho, el número de modelos posibles es 2 elevado al número

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

de proposiciones. En este caso, teníamos 2 proposiciones, entonces 2^2=4 modelos posibles.

La base de conocimiento KB es un conjunto de sentencias conocidas por un agente basado en conocimiento. Este es el conocimiento que la IA proporciona sobre el mundo en forma de sentencias de lógica proposicional que se pueden usar para hacer inferencias adicionales sobre el mundo.

En el caso de la **vinculación** (⊨), si α ⊨ β (α vincula β), entonces en cualquier mundo donde α sea verdadero, β también lo será (García Serrano, 2012). Por ejemplo, si α: "Es un martes de enero" y β: "Es un martes", entonces sabemos que α ⊨ β. Si es cierto que es un martes de enero, también sabemos que es martes. La vinculación es diferente de la implicación. La implicación es un conector lógico entre dos proposiciones. La vinculación, por otro lado, es una relación que significa qué si toda la información en α es verdadera, entonces toda la información en β es verdadera.

La **inferencia** es el proceso de derivar nuevas oraciones a partir de las antiguas. Existen múltiples formas de inferir nuevos conocimientos basados en el conocimiento existente. Primero, considérese el algoritmo de verificación de modelo (Model Checking).

• Para determinar si KB ⊨ α (en otras palabras, responde a la pregunta: "¿se puede concluir que α es cierto en base a nuestra base de conocimientos")

1. Se debe enumerar todos los modelos posibles.

2. Si en todos los modelos donde KB es verdadero, α también lo es, entonces KB vincula α (KB ⊨ α).

Considérese el siguiente ejemplo:

P: Es un miércoles. Q: Está lloviendo. R: Patricio irá a correr. KB: (P ∧ ¬Q) → R (en palabras, P y no Q implica R) P (P es verdadera) ¬Q (Q es falsa) Consulta: R (Queremos saber si R es verdadera o falsa; ¿KB ⊨ R?)

Para responder a la consulta utilizando el algoritmo de verificación de modelos, enumeramos todos los modelos posibles (Tabla 2-3).

Tabla 2-3 Modelos posibles

P	Q	R	KB
Falso	Falso	Falso	
Falso	Falso	verdadero	

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

P	Q	R	KB
Falso	verdadero	Falso	
Falso	verdadero	verdadero	
verdadero	Falso	Falso	
verdadero	Falso	verdadero	
verdadero	verdadero	Falso	
verdadero	verdadero	verdadero	

Elaborado por los Autores

Luego, se revisa cada modelo y verificamos si es cierto dada nuestra Base de conocimiento.

Primero, en nuestra KB, sabemos que **p** es verdadera. Así, podemos decir que la KB es falsa en todos los modelos donde p no es verdadera.

Tabla 2-4 Base de conocimiento en base a P

P	Q	R	KB
Falso	falso	Falso	Falso
Falso	falso	verdadero	Falso
Falso	verdadero	Falso	Falso
Falso	verdadero	verdadero	Falso
verdadero	falso	Falso	Verdadero
verdadero	falso	verdadero	Verdadero
verdadero	verdadero	Falso	Verdadero
verdadero	verdadero	verdadero	Verdadero

Elaborado por los Autores

Luego, de manera similar, en nuestra KB, sabemos que Q es falsa. Así, podemos decir que la KB es falsa en todos los modelos donde Q es verdadera.

Tabla 2-5 Base de conocimiento en base a Q

P	Q	R	KB
Falso	falso	Falso	Falso
Falso	falso	verdadero	Falso
Falso	verdadero	Falso	Falso
Falso	verdadero	verdadero	Falso

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

P	Q	R	KB
verdadero	falso	Falso	
verdadero	falso	verdadero	
verdadero	verdadero	Falso	Falso
verdadero	verdadero	verdadero	Falso

Elaborado por los Autores

Finalmente, nos quedamos con dos modelos. En ambos, **P** es verdadera y **Q** es falsa. En un modelo **R** es verdadero y en el otro **R** es falso. Debido a que $(P \wedge \neg Q) \rightarrow R$ está en nuestra KB, sabemos que en el caso de que P sea verdadera y Q sea falsa, R debe ser verdadera. Por lo tanto, decimos que nuestro KB es falso para el modelo donde R es falso y verdadero para el modelo donde R es verdadero.

Tabla 2-6 Base de conocimiento en base a R

P	Q	R	KB
Falso	falso	Falso	Falso
Falso	falso	verdadero	Falso
Falso	verdadero	Falso	Falso
Falso	verdadero	verdadero	Falso
verdadero	falso	Falso	Falso
verdadero	falso	verdadero	Verdadero
verdadero	verdadero	Falso	Falso
verdadero	verdadero	verdadero	Falso

Elaborado por los Autores

Mirando esta tabla, solo hay un modelo donde nuestra base de conocimiento es verdadera. En este modelo, se ve que **R** también es verdadero. Según nuestra definición de vinculación, si r es verdadera en todos los modelos donde KB es verdadera, entonces $KB \vDash R$.

2.1.2 Lógica de primer orden

La lógica de primer orden o lógica proposicional, aunque eficaz para representar conocimientos simples y manejar proposiciones aisladas, enfrenta limitaciones significativas en escenarios más complejos. Su estructura binaria y la falta de conexiones entre proposiciones restringen su capacidad para capturar relaciones y matices detallados en el conocimiento.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Ante estas limitaciones se desarrolló la lógica de primer orden, una evolución de la lógica proposicional. Esta forma lógica más general permite representar todos los detalles expresados en las sentencias de manera más rica y precisa. La lógica de primer orden es capaz de expresar relaciones, cuantificadores, y reglas más complejas, lo que la hace adecuada para representar conocimiento en entornos donde la información es más elaborada y las interacciones entre objetos son más sofisticadas.

En el ámbito de la inteligencia artificial, la lógica de primer orden juega un rol fundamental, proporcionando una base sólida para el desarrollo de sistemas inteligentes. Su capacidad para representar conocimientos de manera detallada y estructurada la convierte en una herramienta indispensable para abordar los desafíos de representación del conocimiento en IA.

La lógica de primer orden se destaca especialmente en entornos de incertidumbre, donde los agentes inteligentes deben operar con información fragmentada o incompleta (Gelfond & Kahl, 2014). Esta lógica ofrece un marco robusto para la interpretación y el manejo de estas situaciones, respaldado por un conjunto de axiomas y reglas bien definidas.

La lógica de primer orden se convierte en un pilar esencial para la inteligencia artificial, proporcionando una estructura para la representación del conocimiento que es fundamental para el desarrollo de comportamientos inteligentes.

En la lógica de primer orden, la construcción del conocimiento se basa en tres elementos fundamentales:

Constantes: Representan los objetos del conocimiento, que pueden ser personas, objetos físicos o conceptos abstractos. Son los elementos básicos sobre los que se construye el conocimiento.

Predicados: Son relaciones, cualidades o atributos que se aplican a los objetos del conocimiento. Permiten expresar cómo se relacionan entre sí o qué características poseen.

Funciones: Las funciones en la lógica de primer orden son referencias indirectas a otros elementos del conocimiento. Pueden utilizarse para representar transformaciones o cálculos sobre los objetos y predicados.

Estos tres componentes trabajan en conjunto para representar de manera precisa y estructurada el conocimiento en la lógica de primer orden, convirtiéndose en una herramienta poderosa para la representación y el razonamiento en la inteligencia artificial.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

En la lógica de primer orden, las constantes se utilizan para representar objetos individuales en un dominio, estos objetos deben ser relevantes en el contexto del conocimiento que estamos representando.

Ejemplos de constantes podrían ser:

- Juan
- Guillermina
- Bicicleta
- Automóvil

Cada una de estas constantes se refiere a un objeto específico en el dominio de conocimiento y puede utilizarse en sentencias lógicas para expresar relaciones, propiedades o hechos que involucran a estos objetos (Brachman & Levesque, 2004). Las constantes son una parte fundamental de la representación del conocimiento en la lógica de primer orden.

En la lógica de primer orden, los predicados se utilizan para expresar relaciones entre objetos o para definir propiedades de esos objetos. Los predicados representan afirmaciones sobre el mundo y se combinan con las constantes para formar frases atómicas que describen hechos o relaciones específicas en un dominio de conocimiento.

Como ejemplo, se tiene el predicado "persona" que se utiliza para definir la propiedad de ser una persona. Luego, se aplican este predicado a las constantes "Guillermina" y "Juan" para afirmar que Guillermina y Juan son personas. Estas afirmaciones se expresan como frases atómicas en la lógica de primer orden:

Persona (Guillermina)

Persona (Juan)

Estas frases atómicas representan hechos concretos sobre el dominio de conocimiento y pueden utilizarse en inferencias y razonamientos lógicos dentro de un sistema de lógica de primer orden.

De las expresiones anteriores la función es Persona, y el argumento Guillermina y Juan. Se puede indicar que ellos se conocen

Conoce(Juan, Guillermina) expresa: Juan conoce a Guillermina

Conoce(Guillermina, Juan) expresa: Guillermina conoce a Juan

También se puede expresar:

Pasea_en(Juan, bicicleta) expresa: Juan pasea en la bicicleta

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Conduce(Guillermina, Automóvil) expresa: Guillermina conduce el automóvil

Si se aplica de la lógica proposicional los operadores booleanos se pueden construir oraciones más complejas. De esta manera, se toman dos o más frases atómicas y con los conectivos se construye una oración compuesta.

Conoce(Juan, Guillermina) $\wedge$ Conoce(Guillermina, Juan)

expresa: Juan conoce a Guillermina y Guillermina conoce a Juan

Pasea_en(Juan, Bicicleta) $\vee$ Pasea_en(Juan, Automóvil)

expresa: Juan pasea en bicicleta o Juan pasea en automóvil

Conoce(Juan, Guillermina) $\wedge$ $\neg$ Conoce(Guillermina, Juan)

expresa: Juan conoce a Guillermina o Guillermina no conoce a Juan

Conoce(Juan, Guillermina) $\rightarrow$ Conoce(Guillermina, Juan)

expresa: Si Juan conoce a Guillermina entonces Guillermina conoce a Juan

Conoce(Juan, Guillermina) $\leftrightarrow$ Conoce(Guillermina, Juan)

expresa: Juan conoce a Guillermina si Guillermina conoce a Juan

En la lógica de primer orden, las **variables** son herramientas poderosas que permiten representar información de manera abstracta y general, lo que facilita la formulación de reglas y relaciones que son válidas para una amplia gama de objetos sin especificar los objetos en sí. Esto es especialmente útil en la inteligencia artificial y la representación del conocimiento, donde no siempre se conoce de antemano la identidad exacta de los objetos involucrados.

Conoce(x, Guillermina) $\rightarrow$ Conoce(Guillermina, x)

Hermano(x, y) $\rightarrow$ Padre(z, x) $\wedge$ Padre(z, y)

Los predicados en la lógica de primer orden tienen un valor de veracidad que depende de los términos específicos que se les asignen. Esto significa que un predicado puede ser verdadero para ciertos términos y falso para otros, lo que le brinda una capacidad de expresión muy flexible.

Esta característica es fundamental para la representación precisa y abstracta del conocimiento en la inteligencia artificial y otros campos.

Los predicados permiten asignar cualidades abstractas o relaciones entre objetos o conceptos. Por ejemplo, el predicado "Conoce (x, y)" podría representar la relación de conocer entre dos personas, donde "x" y "y" son variables que pueden tomar cualquier valor de individuos. Esto permite representar declaraciones como "Juan conoce a María"

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

(donde "x" toma el valor de Juan y "y" toma el valor de María) o "Xavier conoce a Silvana" (donde "x" toma el valor de Xavier y "y" toma el valor de Silvana).

La lógica de primer orden se basa en **reglas de inferencia** sólidas para el razonamiento lógico. Esta lógica se utiliza para deducir consecuencias verdaderas a partir de un conjunto de predicados que se asumen como lógicamente verdaderos, generalmente denominados axiomas o hechos iniciales.

Las reglas de inferencia son principios lógicos que determinan cómo se pueden derivar nuevas afirmaciones o conclusiones de las afirmaciones o premisas existentes. Estas reglas aseguran que las inferencias realizadas sean válidas y que las conclusiones sean verdaderas si las premisas lo son. Algunos ejemplos de reglas de inferencia en la lógica de primer orden incluyen el modus ponens, modus tollens, la introducción de cuantificadores y otras.

2.1.3 Cuantificadores

Los cuantificadores son herramientas esenciales en la lógica de primer orden para expresar afirmaciones que involucran variables y establecer si son verdaderas para todos los elementos del dominio o al menos para algún elemento.

El **cuantificador universal (∀)** se utiliza para expresar una propiedad que es verdadera para todos los elementos del dominio (Van Harmelen et al., 2008). Por ejemplo, ∀x P(x) puede leerse como "Para todo x, P(x) es verdadero". Indica que la propiedad P(x) es verdadera para cada elemento x en el dominio.

El **cuantificador existencial (∃)** se utiliza para expresar una propiedad que es verdadera para al menos un elemento del dominio (Van Harmelen et al., 2008). Por ejemplo, ∃x P(x) puede leerse como "Existe al menos un x para el cual P(x) es verdadero". Indica que la propiedad P(x) es verdadera para al menos un elemento x en el dominio.

Estos cuantificadores se combinan con predicados para formar expresiones lógicas más complejas y precisas. Por ejemplo, se pueden usar en afirmaciones como ∀x (P(x) → Q(x)) que se lee como "Para todo x, si P(x) es verdadero, entonces Q(x) también es verdadero". O ∃x (P(x) ∧ Q(x)) que se lee como "Existe al menos un x para el cual P(x) y Q(x) son verdaderos".

La cuantificación es fundamental en la representación del conocimiento y el razonamiento en la inteligencia artificial y otras áreas de la informática. Permite expresar propiedades generales sobre objetos y relaciones entre ellos, lo que es importante para

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

modelar situaciones del mundo real de manera precisa y eficiente.

Ejemplos del uso de cuantificadores:

A todos los niños les agrada los dulces.

$\forall x \ \text{Niño}(x) \rightarrow \text{Agrada}(x, \text{Dulces})$

Algunos estudiantes se matriculan en inteligencia artificial.

$\exists x \ \text{Estudiante}(x) \wedge \ \text{Matricula}(x, \text{IA})$

Hay un cocinero que cocina para todas las personas del barrio que no pueden cocinar ellos mismos.

$\exists x \ \text{Cocinero}(x) \wedge \ \forall y \ \text{Persona}(y) \wedge \ \neg \ \text{Cocina}(y,y) \ \rightarrow \ \text{Cocina}(x,y)$

Para que dos personas sean hermanos.

$\forall x,y \ \textit{Hermano}(x,y) \rightarrow \ \exists z \ \textit{Padre}(z,x) \ \wedge \ \textit{Padre}(z,y)$

2.1.4 Razonamiento

El razonamiento es un proceso esencial en la toma de decisiones y la resolución de problemas tanto en humanos como en sistemas de inteligencia artificial. El razonamiento lógico y las teorías de razonamiento automatizado son esenciales en campos como la inteligencia artificial, la programación, la verificación de la corrección de programas y muchos otros.

En el contexto de la lógica de predicados, las inferencias son el proceso de derivar nuevas sentencias lógicas o conclusiones a partir de un conjunto dado de sentencias conocidas o premisas. Estas inferencias se realizan mediante reglas de inferencia que garantizan la validez y la coherencia de las conclusiones.

Algunas de las reglas de inferencia más importantes en lógica de predicados incluyen:

Modus Ponens: Si tenemos una afirmación condicional ($p \rightarrow q$) y sabemos que la afirmación antecedente (p) es verdadera, entonces podemos concluir que la afirmación consecuente (q) también es verdadera.

Modus Tollens: Si tenemos una afirmación condicional ($p \rightarrow q$) y sabemos que la afirmación consecuente (q) es falsa, entonces podemos concluir que la afirmación antecedente (p) también es falsa.

Silogismo Hipotético: Si tenemos dos afirmaciones condicionales ($p \rightarrow q$) y ($q \rightarrow r$), entonces podemos inferir una nueva afirmación condicional ($p \rightarrow r$).

Silogismo Disyuntivo: Si tenemos una afirmación que es una disyunción ($p \vee q$)

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

y sabemos que una de las disyunciones (p) es falsa, entonces podemos concluir que la otra disyunción (q) es verdadera.

Cuantificación Universal y Existencial: Utilizando los cuantificadores (∀ y ∃), podemos realizar inferencias basadas en propiedades que se aplican a todos los elementos del dominio o al menos a algunos de ellos.

Estas reglas de inferencia son fundamentales en la lógica de predicados y se utilizan para derivar conclusiones válidas a partir de premisas dadas. La aplicación precisa de estas reglas es esencial para el razonamiento lógico y la toma de decisiones en sistemas inteligentes.

a) Modus ponens.- Es utilizada, en los sistemas basados en conocimiento; si sabemos que una implicación y su antecedente son verdaderos, entonces el consecuente también es verdadero, entonces se establece que:

Si se tiene una afirmación condicional (p → q) y se sabe que la afirmación antecedente (p) es verdadera, entonces se puede concluir que la afirmación consecuente (q) también es verdadera.

Ejemplo 1

“Hay luz en la lámpara” representa a la sentencia p si esta es verdadera. Si el condicional siguiente es verdadero: “Si hay luz en la lámpara entonces el interruptor está encendido”. La sentencia q “el interruptor esta encendido” es verdadero. En otras palabras, el uso de esta regla, permite obtener sintácticamente nuevo conocimiento del viejo conocimiento. En este ejemplo se puede pensar en p como el antecedente y en q como el consecuente.

Ejemplo 2

Premisa 1: Si llueve, entonces la calle estará mojada.

Premisa 2: Está lloviendo.

Conclusión: Por lo tanto, la calle estará mojada.

En este caso, se sabe que si está lloviendo (p es verdadero), entonces la afirmación condicional "Si llueve, entonces la acera estará mojada" nos dice que la acera estará mojada (q es verdadero).

Ejemplo 3

Premisa 1: Si estudias para el examen (p), entonces obtendrás una buena calificación (q).

Premisa 2: Estás estudiando para el examen (p es verdadero).

Conclusión: Por lo tanto, obtendrás una buena calificación (q es verdadero).

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

En este caso, la afirmación condicional establece que si estudias para el examen (p), entonces obtendrás una buena calificación (q). La premisa 2 nos dice que estás estudiando para el examen, lo que significa que p es verdadero. Según el Modus Ponens, podemos concluir que obtendrás una buena calificación (q es verdadero) debido a que se cumple la condición establecida en la premisa 1.

b) **Modus tollens.-** Es una regla de inferencia lógica que opera de manera contraria al Modus Ponens. En el Modus Tollens, se parte de una afirmación condicional (p → q), se sabe que la afirmación consecuente (q) es falsa, entonces se puede concluir que la afirmación antecedente (p) también debe ser falsa.

El Modus Tollens es una herramienta importante en la lógica y el razonamiento deductivo para evaluar las consecuencias de afirmaciones condicionales y su negación.

Ejemplo1

Retomando el ejemplo anterior en el cual se dice que hay luz en la lámpara y el interruptor está encendido. Si se expresa el condicional la lámpara está apagada por lo tanto el interruptor está apagado, esto es verdadero, para nuestra forma lógica de pensar, pero para el caso que estamos tratando q “el interruptor está apagado” es falso, por lo tanto, la premisa p “la lámpara está apagada” también es falsa. Se puede observar que es el enfoque contradictorio del Modus Ponens.

Ejemplo 2

Premisa 1: Si llueve, la calle estará mojada (p → q).

Premisa 2: La calle no está mojada (¬q).

Conclusión: Por lo tanto, no está lloviendo (¬p).

En este caso, se sabe que la calle no está mojada (¬q), se concluye que no está lloviendo (¬p) utilizando el Modus Tollens, debido a que la premisa inicial establece que si llueve (p), la calle estará mojada (q). Dado que la calle no está mojada (¬q), podemos inferir que no está lloviendo (¬p).

Ejemplo 3

Premisa 1: Si el estudiante estudia para el examen, aprobará (p → q).

Premisa 2: El estudiante no aprobó el examen (¬q).

Conclusión: Por lo tanto, el estudiante no estudió para el examen (¬p).

En este caso, si se sabe que el estudiante no aprobó el examen (¬q), se puede concluir que el estudiante no estudió para el examen (¬p) utilizando el Modus Tollens.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

La premisa inicial establece que si el estudiante estudia para el examen (p), aprobará (q). Dado que el estudiante no aprobó (¬q), puede inferir que no estudió para el examen (¬p).

c) ***Resolución.-*** Esta regla permite la creación de una contradicción a través de la negación de la sentencia original, que permite demostrar que la sentencia original es verdadera. La resolución es un método de inferencia utilizado en lógica para determinar la verdad o falsedad de una afirmación basándose en un conjunto de cláusulas de conocimiento y una consulta

Ejemplo 1

Si (A v B) es verdadero y **(**¬B v C) es verdadero, entonces (A v C) también es verdadero.

Las sentencias expresan lo siguiente: A = “la lámpara está apagada”, B = “el interruptor está apagado”, C = “el enchufe está desconectado”. Si se niega B = “el interruptor no está apagado”. Entonces A es verdadero, y B es falso, entonces C es verdadero.

Ejemplo 2

Premisa 1: Si llueve, entonces la calle estará mojada (p → q).

Premisa 2: La calle no está mojada (¬q).

Consulta: ¿Está lloviendo? (¬p)

Para resolver esto, primero convertimos las premisas a cláusulas:

Premisa 1 (Cláusula 1): ¬p ∨ q

Premisa 2 (Cláusula 2): ¬q

Ahora, aplicamos la resolución:

(¬p ∨ q) ∧ (¬q) [Conjuntamos las cláusulas]

(¬p ∧ ¬q) ∨ (q ∧ ¬q) [Distribución]

(¬p ∧ ¬q) ∨ F [q ∧ ¬q es F (falso)]

(¬p ∧ ¬q) [Identidad ((¬p ∧ ¬q) ∨ F = (¬p ∧ ¬q))]

La resolución nos lleva a la conclusión de que ¬p es verdadero, lo que significa que no está lloviendo.

Ejemplo 3

Premisa 1: Si el estudiante estudia para el examen, aprobará (p → q).

Premisa 2: El estudiante no aprobó el examen (¬q).

Consulta: ¿El estudiante estudió para el examen? (¬p)

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Convertimos las premisas a cláusulas:

Premisa 1 (Cláusula 1): ¬p ∨ q

Premisa 2 (Cláusula 2): ¬q

Aplicamos la resolución:

(¬p ∨ q) ∧ (¬q) [Conjuntamos las cláusulas]

(¬p ∧ ¬q) ∨ (q ∧ ¬q) [Distribución]

(¬p ∧ ¬q) ∨ F [q ∧ ¬q es F (falso)]

(¬p ∧ ¬q) [Identidad ((¬p ∧ ¬q) ∨ F = (¬p ∧ ¬q))]

La resolución nos lleva a la conclusión de que ¬p es verdadero, lo que significa que el estudiante no estudió para el examen.

En estos ejemplos, la resolución nos permite llegar a la misma conclusión que obtuvimos utilizando el Modus Tollens.

d) Deducción.- Es un tipo de razonamiento que procede lógicamente de lo universal a lo particular, a partir de premisas verdaderas garantiza conclusiones verdaderas. Es la base de la lógica proposicional y de predicados.

Figura 2.1: Tamaño de recipientes

Elaborado por los Autores

Ejemplo 1:

Se tienen tres recipientes (figura 1.17) de diferente tamaño x, y, z.

Si ***x*** es mayor que ***y*** y ***y*** es mayor que ***z***

entonces se puede deducir que ***x*** es mayor que ***z***

Utilizando los cuantificadores esto se puede expresar como.

$\forall x, \forall y, \forall z, [Mayor\ (x, y) \wedge Mayor\ (y, z) \rightarrow Mayor\ (x, z)]$

Ejemplo 2

Premisa 1: Todos los hombres son mortales. (Premisa general)

∀x(Hombre(x)→Mortal(x))

Premisa 2: Platón es un hombre. (Premisa específica)

Hombre(Platón)

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Conclusión: Por lo tanto, Sócrates es mortal. (Deducción basada en las dos premisas anteriores y la regla lógica de que si todos los hombres son mortales y Sócrates es un hombre, entonces Sócrates también es mortal).

Hombre(Platón))→Mortal(Platón)

Ejemplo 3

Premisa 1: a=b (Premisa)

Premisa 2: b=c (Premisa)

Conclusión: Por lo tanto, a=c (Deducción basada en las dos premisas anteriores y la propiedad transitiva de la igualdad en las matemáticas, que establece que si a=b y b=c, entonces a=c).

(a=b ∧ b=c)→(a=c)

En estos ejemplos, la deducción implica llegar a una conclusión lógica o matemática basada en premisas o información previamente establecida. La deducción es un proceso importante en la lógica, donde se utilizan reglas y principios para derivar conclusiones válidas a partir de información conocida.

e) Abducción.- Es un método de razonamiento utilizado en la generación de explicaciones, el cual no garantiza que se puedan lograr conclusiones verdaderas, por lo tanto no constituye un método sólido de inferencia (Kowalski, 2011).

En términos generales se puede expresar:

Si la sentencia (A→B) es verdadera y B es verdadera,

entonces A es posiblemente verdadera.

En la abducción, se parte de una conclusión y se busca identificar las circunstancias que la justificarían. Es decir, se intenta hallar una razón o causa que explique por qué la conclusión es correcta. Es comúnmente utilizado en inteligencia artificial y razonamiento basado en casos, así como en diagnóstico médico y resolución de problemas.

Ejemplo 1

Si se tiene una funda con esferográficos de color azul, y junto a la funda están unos esferográficos de color azul, se deduce que proceden estos esferográficos de la funda, esto último posiblemente puede ser verdadero.

Ejemplo 2

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Supóngase que existe una persona llamada Luis que siempre llega tarde a las reuniones. Un día, Luis llega tarde a una reunión nuevamente. Se puede utilizar la abducción para generar una explicación probable de por qué Luis llega tarde. Las observaciones son:

Observación1: Luis llegó tarde a la reunión.

Observación 2: Luis tiene un largo trayecto en automóvil para llegar al lugar de la reunión.

Observación 3: Hubo un accidente en la carretera principal.

Hipótesis 1: Luis llegó tarde a la reunión por el accidente en la carretera principal.

Hipótesis 2: Luis llegó tarde a la reunión porque se levanta muy tarde.

Hipótesis 3: Luis llegó tarde a la reunión por ir a dejar a sus hijos al colegio.

Ahora, se puede realizar un razonamiento abductivo para generar una explicación, dada esta información, se podría abducir la siguiente explicación probable:

Luis llegó tarde a la reunión (O1) porque (H1) hubo un accidente en la carretera principal (O3), lo que causó retrasos en el tráfico y afectó su largo trayecto en automóvil (O2).

En este ejemplo, la abducción permite generar una explicación probable (el accidente en la carretera) para la hipótesis (porque, Luis llegó tarde a la reunión) basada en las circunstancias conocidas (largo trayecto de Juan y el accidente en la carretera).

Ejemplo 3

Imagine que usted es un detective y llega a una casa, donde encuentra una ventana rota, una puerta forzada y un escritorio revuelto. Tiene las siguientes observaciones:

Observación 1: La ventana está rota.

Observación 2: La puerta ha sido forzada.

Observación 3: El escritorio del cuarto de estudio está revuelto.

Observación 4: El propietario de la casa informa que algunas joyas han desaparecido.

Ahora, quieres abducir una explicación probable de lo que sucedió. Las

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

hipótesis posibles incluyen:

Hipótesis 1: Un ladrón entró por la ventana rota.

Hipótesis 2: El propietario perdió las joyas y forzó la puerta y revolvió el escritorio para hacer que pareciera un robo.

Hipótesis 3: Hubo un terremoto que causó la ventana rota, la puerta forzada y el escritorio revuelto.

Dado que está buscando una explicación como detective podría usar la hipótesis más probable, para abducir lo siguiente:

Es probable que un ladrón (H1) haya entrado a la casa por la ventana rota (O1), forzado la puerta (O2) y revuelto el escritorio (O3) en busca de las joyas (O4) que luego se llevó.

En este caso, la abducción le ayuda a generar la hipótesis más probable basada en las observaciones en la escena del crimen. Sin embargo, es importante recordar que la abducción no garantiza la verdad absoluta, solo identifica una explicación probable dadas las observaciones disponibles.

f) ***Inducción.-*** La inducción representa un proceso de razonamiento que parte de observaciones específicas para alcanzar conclusiones de carácter más general. En el vasto y complejo campo de la inteligencia artificial, este enfoque se emplea de manera estratégica para elaborar modelos, patrones y reglas, todos ellos fundamentados en datos concretos y observaciones detalladas.

Esta metodología no solo facilita la creación de sistemas predictivos robustos, sino que también dota a las máquinas de la capacidad de tomar decisiones informadas y precisas en escenarios futuros que presenten similitudes con las situaciones observadas. Así, la inducción se convierte en una herramienta esencial para la extracción de conocimiento y aprendizaje automático, permitiendo a los sistemas de IA adaptarse y responder de manera efectiva ante una variedad de situaciones y contextos.

La forma del método inductivo es la siguiente:

Si se conoce que P(a), P(b),..... P(n) son verdaderos,

entonces se puede concluir que ∀x, P(x) es también verdadero.

Se emplea para los estudios experimentales, por ejemplo, cuando se prueban las medicinas se hace en un grupo que tiene ciertas características y luego

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

cuando ha dado resultado se generaliza a la población, pero cuando ya está en el mercado la medicina, hay ciertos pacientes a los cuales les es contraproducente.

En el contexto de los estudios experimentales, como la prueba de medicamentos, se utiliza un enfoque inductivo. Este proceso comienza con la selección de un grupo específico de pacientes que comparten ciertas características, sobre los cuales se realizan pruebas. Los resultados positivos obtenidos en este grupo limitado se extrapolan luego para aplicarlos a la población general. Sin embargo, es importante reconocer que, incluso después de la aprobación y distribución de un medicamento, pueden surgir casos individuales de efectos adversos. Este fenómeno destaca la importancia de la vigilancia continua y del análisis de datos post-lanzamiento, un área donde la inteligencia artificial puede jugar un rol significativo, procesando y analizando grandes volúmenes de datos para identificar patrones ocultos y predecir posibles contraindicaciones en subgrupos específicos de pacientes.

La inducción en el ámbito de la inteligencia artificial abarca un espectro amplio de técnicas y métodos, cada uno con sus aplicaciones y ventajas específicas. Entre estas técnicas se incluyen:

Aprendizaje Supervisado: Donde los modelos de IA aprenden a partir de conjuntos de datos etiquetados, mejorando su capacidad para hacer predicciones o clasificaciones precisas.

Aprendizaje No Supervisado: Utilizado para analizar datos sin etiquetar, descubriendo estructuras ocultas o patrones sin intervención humana previa (Xiao, 2022).

Minería de Datos: Esencial para descubrir correlaciones y tendencias en grandes bases de datos, lo que permite una toma de decisiones más informada.

Razonamiento Inductivo: Se aplica para formular hipótesis y modelos a partir de observaciones específicas.

Redes Neuronales: Sistemas que imitan la estructura y función del cerebro humano para procesar información de manera eficiente.

Aprendizaje Profundo: Una forma avanzada de aprendizaje automático que utiliza redes neuronales profundas para modelar y resolver problemas complejos.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Cada uno de estos enfoques aporta una dimensión única al campo de la IA, permitiendo a los sistemas aprender y adaptarse de manera efectiva en una variedad de contextos y aplicaciones.

A continuación, tenemos algunos ejemplos:

Ejemplo 1

Clasificación de correos electrónicos como spam o no spam

El proceso empieza con la recopilación de un conjunto de datos que contiene una colección de correos electrónicos que se etiquetan previamente como "spam" o "no spam". Cada correo electrónico en el conjunto de datos se representa como un conjunto de características, como las palabras que contiene y otros atributos relevantes.

Luego el conjunto de datos de entrenamiento se utilizzan para enseñar a nuestro algoritmo de aprendizaje supervisado a reconocer patrones que distingan entre correos electrónicos de spam y no spam. El algoritmo ajusta un modelo matemático que relaciona las características de entrada (contenido del correo electrónico) con las etiquetas de salida (spam o no spam).

Durante el proceso de entrenamiento, el algoritmo de aprendizaje inducirá reglas o relaciones a partir de los ejemplos de entrenamiento. Por ejemplo, podría aprender que, si un correo electrónico contiene ciertas palabras clave como "oferta", "ganador" y "gratis", es más probable que sea spam.

Después de entrenar el modelo, se valida utilizando un conjunto de datos de prueba independiente para verificar su capacidad para clasificar correctamente correos electrónicos desconocidos. Si el modelo no tiene un rendimiento satisfactorio, se puede ajustar los parámetros del algoritmo o recopilar más datos de entrenamiento para mejorar su precisión.

Una vez que el modelo está entrenado y validado, se puede usarlo para clasificar automáticamente nuevos correos electrónicos. Cuando llega un nuevo correo electrónico, el modelo evalúa sus características y predice si es spam o no spam en función de las reglas y patrones que ha aprendido.

Este es un ejemplo simplificado, debido a que en la práctica, los algoritmos de aprendizaje supervisado pueden ser mucho más complejos y manejar una variedad de características y técnicas de procesamiento de datos. La clave de la

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

inducción en este caso es que el modelo aprende a partir de ejemplos previos para generalizar y hacer predicciones precisas sobre nuevos datos sin etiquetar.

Ejemplo 2

Clasificación de dígitos escritos a mano

Se desea construir un sistema de reconocimiento de dígitos del 0 al 9 escrito a mano, de forma que el sistema pueda identificar automáticamente el dígito contenido en una imagen.

Se reúne un conjunto de datos de entrenamiento que incluye miles de imágenes de dígitos manuscritos, donde cada imagen está marcada con la identificación del dígito que representa, como "0", "1", "2", "3", ..., "9".

Cada imagen se transforma en un conjunto de características numéricas que capturan distintas propiedades de la imagen, como la intensidad de los píxeles y la forma de los trazos. Estas características se utilizan como la entrada de nuestro modelo.

Este conjunto de datos de entrenamiento, junto con las etiquetas correspondientes, se emplea para entrenar un modelo de aprendizaje supervisado, como una red neuronal o un clasificador basado en máquinas de soporte vectorial (SVM). El modelo aprenderá a identificar patrones en las características que permitan distinguir entre los diversos dígitos.

En el proceso de entrenamiento, el modelo inducirá reglas o pesos óptimos a cada característica, lo que le permitirá realizar una clasificación precisa de los dígitos.

El modelo se somete a una validación utilizando un conjunto de datos de prueba que no fue empleado durante el proceso de entrenamiento. En caso de que el modelo no presente un desempeño satisfactorio, se procede a ajustar sus parámetros o a considerar estrategias adicionales, como aumentar el tamaño del conjunto de datos de entrenamiento o realizar modificaciones en las características extraídas (McCarthy, 1981).

Una vez que el modelo ha sido entrenado y validado, está listo para realizar predicciones en imágenes nuevas y desconocidas de manera automática. El modelo examina las características de la imagen y emite una predicción sobre el dígito que esta representa.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Ejemplo 3

Procesamiento del lenguaje natural (PNL)

Suponga que se desea construir un sistema que pueda identificar el idioma en el que está escrito un texto.

Se comienza recopilando un conjunto de textos en varios idiomas diferentes, y cada texto está etiquetado con el idioma en el que está escrito.

Se Utiliza este conjunto de datos para entrenar un modelo de aprendizaje automático, como un clasificador basado en palabras o caracteres, para identificar patrones que distingan entre los diferentes idiomas.

Durante el proceso de entrenamiento, el algoritmo de aprendizaje induce reglas basadas en las palabras, caracteres u otras características relevantes de cada texto en diferentes idiomas. Por ejemplo, podría aprender que ciertas combinaciones de caracteres son comunes en un idioma específico, pero raras en otros.

Se evalúa el rendimiento del modelo utilizando un conjunto de datos de prueba independiente y se ajusta la configuración si es necesario para mejorar su precisión.

Una vez que el modelo está entrenado y validado, se puede usar para predecir el idioma de un nuevo texto en función de las reglas y patrones que ha aprendido. Por ejemplo, si un texto tiene muchas palabras que son típicas del español, el modelo lo clasificará como "español".

La inducción en este caso implica aprender reglas y patrones a partir de datos de entrenamiento para tomar decisiones informadas sobre datos no vistos anteriormente.

2.2 Reglas de producción

En el ámbito de la inteligencia artificial, las bases de conocimiento se estructuran mediante cláusulas de Horn. Las cláusulas de Horn reciben su nombre del lógico Alfred Horn, quien las identificó como una forma especial de expresión lógica. Estas cláusulas se articulan |a través de una estructura de implicación, en la cual la premisa se compone de una conjunción de literales positivos (que pueden ser hechos o condiciones), y la conclusión es un solo literal positivo. Para llevar a cabo procesos inferenciales, es común emplear algoritmos de encaminamiento, tanto en modalidades hacia adelante como hacia

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

atrás.

Son particularmente significativas en la inteligencia artificial debido a su eficiencia en el procesamiento computacional. Esta estructura facilita enormemente el diseño de algoritmos de inferencia, debido a que permite simplificar los procesos de búsqueda y de resolución de problemas. Además, las cláusulas de Horn son la base para muchos lenguajes de programación lógica, como Prolog, que son ampliamente utilizados en la construcción de sistemas basados en conocimientos y en la resolución de problemas en IA. Su simplicidad y potencia hacen que las cláusulas de Horn sean ideales para representar reglas y hechos en sistemas expertos, permitiendo a estos sistemas realizar inferencias lógicas de manera eficiente y efectiva.

Los sistemas basados en reglas representan un destacado paradigma en la representación del conocimiento debido a su simplicidad y similitud con el razonamiento humano. Estos sistemas utilizan el formato IF-THEN (SI-ENTONCES), que es una representación de la cláusula de Horn. En este formato, el "IF" actúa como el antecedente o premisa, mientras que el "THEN" sirve como el consecuente o conclusión. Un ejemplo práctico sería: SI el lugar está sucio, ENTONCES proceda a limpiar.

Los sistemas basados en reglas emplean el modus ponens para gestionar los hechos conocidos. Estos hechos, que son declaraciones aceptadas como verdaderas, forman la memoria de trabajo del sistema. Durante el proceso de inferencia, estas declaraciones y reglas se utilizan conjuntamente para derivar conclusiones.

La unificación es un proceso importante en los sistemas de reglas de producción, representando un mecanismo fundamental para lograr la correspondencia entre diferentes elementos de conocimiento. En esencia, la unificación implica la comparación y combinación de términos, variables o patrones dentro de las reglas y hechos para encontrar un ajuste o coincidencia común. Este proceso no solo permite verificar si dos expresiones son equivalentes o compatibles, sino que también facilita la generación de conclusiones lógicas.

Por ejemplo, en un sistema de diagnóstico médico, la unificación podría utilizarse para combinar síntomas reportados por un paciente (como fiebre o tos) con patrones conocidos de enfermedades, permitiendo al sistema inferir posibles diagnósticos. Además, la unificación es fundamental en la generalización y especialización de las reglas, permitiendo a los sistemas de IA adaptar su conocimiento a nuevos casos o

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

situaciones específicas, lo que es necesario para la flexibilidad y la capacidad de aprendizaje de estos sistemas.

En el ámbito del razonamiento computacional, se destacan principalmente dos enfoques estratégicos: el encaminamiento hacia adelante y el encaminamiento hacia atrás. El encaminamiento hacia adelante comienza con un conjunto exhaustivo de datos conocidos, desde los cuales se desarrolla progresivamente hacia una solución. Este método es especialmente eficaz en situaciones donde los datos fluyen de manera secuencial y sistemática, permitiendo al sistema formular hipótesis y llegar a conclusiones paso a paso. Se emplea comúnmente en sistemas expertos para realizar diagnósticos o en procesos de toma de decisiones donde la información se acumula y se elabora gradualmente.

Por otro lado, el encaminamiento hacia atrás adopta una aproximación inversa. Aquí, se inicia con una hipotética solución o conclusión y luego se busca retroactivamente evidencia o datos que la sustenten. Esta técnica es particularmente útil en contextos donde el objetivo final es claro, pero el camino para alcanzarlo no lo es. Se utiliza en la solución de problemas complejos, como en la programación lógica, donde se definen los objetivos deseados y el sistema trabaja hacia atrás para encontrar las condiciones que deben cumplirse para alcanzar esos objetivos. Esta estrategia es eficaz para reducir el espacio de búsqueda y para centrarse en los aspectos relevantes de un problema, lo que resulta en una solución más eficiente y dirigida.

2.2.1 Encaminamiento hacia adelante

Dentro del marco de los sistemas basados en conocimiento, se opera con un conjunto de datos previamente conocidos. Estos datos se analizan en relación con cada una de las reglas estructuradas en formato IF-THEN. El objetivo es determinar si los datos cumplen con las premisas (IF) de alguna de estas reglas. Para este propósito, se emplea la técnica de unificación. En situaciones donde múltiples reglas resultan ser compatibles, se opta por aquella con la mayor prioridad asignada. Una vez seleccionada, la regla se ejecuta, derivando nuevos hechos (THEN) (Feigenbaum & others, 1977). Estos hechos recién derivados pueden, a su vez, ser empleados por otras reglas, generando así una cascada de derivaciones adicionales. El motor de inferencia es el componente encargado de interpretar y evaluar estos hechos dentro de la base de conocimiento, conduciendo al sistema hacia una respuesta coherente.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Para ilustrar la forma en que trabaja se presenta el siguiente ejemplo con un conjunto de reglas que trata sobre animales:

REGLA 1 IF mamífero = si AND
carnívoro = si AND
color = leonado AND
manchas_oscuras = si
THEN animal = onza
REGLA 2 IF posee_pelo = si AND
puede_dar_leche = si
THEN mamífero= si
REGLA 3 IF come_carne = si AND
dientes_puntiagudos = si AND
garras = si AND
ojos_adelante = si
THEN carnívoro = si
REGLA 4 IF posee_pelo = no AND
tiene_plumas = si AND
puede_volar = si OR
puede_volar = no AND
puede_nadar = si OR
puede_nadar = no AND
pone_huevos = si
THEN pájaro = si
REGLA 5 IF mamífero = si AND
pezuñas = si AND
rumear = si
THEN ungulado = si
REGLA 6 IF mamífero = si AND
carnívoro = si AND
color=leonado AND
rayas_negras = si
THEN animal = tigre
REGLA 7 IF ungulado = si AND
cuello_largo = si AND
patas_largas = si AND
manchas_oscuras = si
THEN animal = jirafa
REGLA 8 IF pájaro = si AND
puede_volar = no AND
cuello_largo = si AND
patas_largas = si AND

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

color = blanco_negro
THEN animal = avestruz
REGLA 9 IF ungulado = si AND
rayas_negras = si AND
color = blanco
THEN animal = cebra
REGLA 10 IF pájaro = si AND
puede_volar = si AND
color = blanco_negro
THEN animal = albatros
REGLA 11 IF pájaro = si AND
puede_volar = no AND
puede_nadar = si AND
color = blanco_negro
THEN animal = pingüino

Se va a probar si se puede determinar el animal al que se refiere dado el conjunto de datos:

posee_pelo = si
puede_dar_leche = si
color=leonado
rayas_negras = si
come_carne = si
dientes_puntiagudos = si
garras =si
ojos_adelante = si

A continuación, se puede observar las ejecuciones necesarias, las reglas que se ajustan a los datos y cuál es la que se aplica y los hechos encontrados.

Tabla 2-7 Fases del proceso de encaminamiento hacia adelante

Número de ejecuciones	Reglas que se ajustan a los datos	Regla Aplicable	Datos aplicables	Hechos encontrados
1	2,3	2	posee_pelo = si puede_dar_leche = si	mamífero = si
2	~~2~~,3	3	come_carne = si dientes_puntiagudos = si garras = si ojos_adelante = si	carnívoro = si
3	~~2,3~~,6	6	mamífero = si carnívoro = si color = leonado	animal = tigre

			rayas_negras = si	
4	~~2,3,6~~	Ninguna		

Elaborado por los Autores

1era Ejecución: Con los datos que se tienen se va a ver cuáles premisas de las reglas se cumplen, encontrando que la regla 2 y 3 se ajustan a los datos dados. La regla aplicable es la 2 por prioridad. Encontrándose el hecho de que se trata de un mamífero. El nuevo hecho pasa a constituirse en un nuevo dato para la búsqueda de las reglas aplicables.

2da Ejecución: Las reglas 2 y 3 se ajustan de nuevo a los datos que se tienen, pero como la regla 2 ya se ejecutó se procede a utilizar la regla 3; derivándose el nuevo hecho que constituye un nuevo dato de que se trata de un animal carnívoro.

3era Ejecución: Las reglas aplicables son la 2,3,6 como las dos primeras ya se utilizaron entonces queda por ejecutarse la regla 6 llegando a determinarse el hecho de que el animal al que se refieren los datos dados es el tigre.

4ta Ejecución: Se vuelve a probar para ver si existe alguna otra regla que se pueda ejecutar, pero como se observa en el cuadro no existe ninguna otra regla; además si hubiese constituiría un error en las premisas de las reglas, pues no existen 2 animales en el mundo con las mismas características.

2.2.2 Encaminamiento hacia atrás

En el enfoque del encaminamiento hacia atrás, el razonamiento se inicia desde la conclusión (THEN) que se busca alcanzar. Inicialmente, no se dispone de datos conocidos. Por ello, basándose en los objetivos presentes en las conclusiones (THEN), se intenta deducir valores a través de cuestionamientos vinculados a las premisas (IF), utilizando las reglas establecidas en la base de conocimientos. Es esencial priorizar estos objetivos para determinar cuál de ellos puede ser satisfecho con las premisas que se van identificando como pertinentes al objetivo en cuestión.

Si todas las premisas de una regla resultan ser verdaderas, se ejecuta dicha regla, derivando así su conclusión y logrando el objetivo propuesto. Este objetivo satisfecho se transforma en un hecho conocido. Sin embargo, si alguna premisa no se cumple, es necesario identificar una regla cuya conclusión (THEN) tenga dicha premisa y, a partir de ahí, trabajar en satisfacer sus premisas (IF) asociadas.

Como aplicación para un mejor entendimiento, con las mismas reglas utilizadas para el encaminamiento hacia adelante se va a proceder a realizar el encaminamiento

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

hacia atrás, siendo el objetivo determinar el animal del que se trata.

1era Ejecución: Se busca todas las reglas que en su conclusión (THEN) mencionen al objetivo animal. Se parte sin datos conocidos.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
Animal	Ninguno	1,6,7,8,9,10,11	1

2da Ejecución: Las reglas que hacen referencia al objetivo animal en las conclusiones son: 1,6,7,8,9,10,11; se procede a probar la primera premisa de la regla 1 mamífero = si, como no existe valor alguno para mamífero en los datos conocidos, se busca una regla que nos permita derivar dicho valor; se encuentra que la regla 2 en su conclusión(THEN) hace referencia a mamífero = si; entonces esta regla se vuelve en la prioridad, y el objetivo cambia a mamífero.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
mamífero animal	Ninguno	2,1,6,7,8,9,10,11	2

3ra Ejecución: Se busca determinar si la primera premisa posee_pelo = si de la regla 2 se encuentra entre los datos conocidos, como no existe se busca determinar si alguna conclusión de las reglas hace referencia, al no encontrarse tampoco el mecanismo de inferencia procede a hacer la pregunta “¿Posee pelo?” al usuario. Ante una respuesta afirmativa, se considera que la premisa es válida y se añade a los datos conocidos.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
mamífero animal	posee_pelo = si	2,1,6,7,8,9,10,11	2

4ta Ejecución: Se busca determinar si la segunda premisa puede_dar_leche = si de la regla 2 se encuentra entre los datos conocidos, como no existe se busca determinar si alguna conclusión (THEN) de las reglas hace referencia, al no encontrarse el mecanismo de inferencia procede a hacer la pregunta “¿Puede dar leche?” al usuario del sistema. La respuesta afirmativa del usuario confirma la premisa, que posteriormente se incorpora a los datos conocidos.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
mamífero animal	posee_pelo = si puede_dar_leche = si	2,1,6,7,8,9,10,11	2

5ta Ejecución: Como las dos premisas de la regla 2 se cumplen entonces se llega a

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

la conclusión de que se trata de un mamífero, por lo tanto el objetivo mamífero pasa a ser parte de los datos conocidos y el objetivo prioritario vuelve a ser animal, siendo la regla 1 la que se debe probar si sus premisas se cumplen.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
animal	posee_pelo = si puede_dar_leche = si mamífero= si	1,6,7,8,9,10,11	1

6ta Ejecución: Se regresa a las reglas aplicables, se procede a probar la primera premisa de la regla 1 mamífero = si, como es parte de los datos conocidos, se procede a determinar si la segunda premisa carnívoro = si se encuentra en los datos conocidos, al no existir valor alguno se busca una regla que nos permita derivar dicho valor; se encuentra que la regla 3 en su conclusión (THEN) hace referencia a carnívoro = si; entonces esta regla se convierte en prioridad, y el objetivo cambia de animal a carnívoro.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
carnívoro animal	posee_pelo = si puede_dar_leche = si mamífero= si	3,1,6,7,8,9,10,11	3

7ma Ejecución: Se busca determinar si la primera premisa come_carne = si de la regla 3 se encuentra entre los datos conocidos como no existe, se busca determinar si alguna conclusión de las reglas hace referencia, al no encontrarse tampoco el mecanismo de inferencia procede a hacer la pregunta "¿Come carne?" al usuario del sistema. La respuesta afirmativa del usuario confirma la premisa, que posteriormente se incorpora a los datos conocidos.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
carnívoro animal	posee_pelo = si puede_dar_leche = si mamífero= si come_carne = si	3,1,6,7,8,9,10,11	3

8va Ejecución: Se busca determinar si la segunda premisa dientes_puntiagudos = si de la regla 3 se encuentra entre los datos conocidos, como no existe se busca determinar si alguna conclusión de las reglas hace referencia a esta premisa, al no encontrarse tampoco el mecanismo de inferencia procede a hacer la pregunta "¿Tiene dientes

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

puntiagudos?" El usuario del sistema responde afirmativamente, validando así la premisa, que se incorpora a la columna de datos conocidos.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
carnívoro animal	posee_pelo = si puede_dar_leche = si mamífero= si come_carne = si dientes_puntiagudos = si	3,1,6,7,8,9,10,11	3

9na Ejecución: Se busca determinar si la tercera premisa garras = si de la regla 3 se encuentra entre los datos conocidos, como no existe se busca determinar si alguna conclusión de las reglas hace referencia a esta premisa, al no encontrarse tampoco el mecanismo de inferencia procede a hacer la pregunta "¿Tiene garras?" al usuario del sistema. La respuesta afirmativa del usuario confirma la premisa, que posteriormente se incorpora a los datos conocidos.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
carnívoro animal	posee_pelo = si puede_dar_leche = si mamífero= si come_carne = si dientes_puntiagudos = si garras =si	3,1,6,7,8,9,10,11	3

10ma Ejecución: Se busca determinar si la cuarta premisa ojos_adelante = si de la regla 3 se encuentra entre los datos conocidos, como no existe se busca determinar si alguna conclusión de las reglas hace referencia a esta premisa, al no encontrarse tampoco el mecanismo de inferencia procede a hacer la pregunta "¿Tiene ojos adelante?" al usuario. La respuesta afirmativa del usuario confirma la premisa, que posteriormente se incorpora a los datos conocidos.

Objetivos	Datos conocidos	Reglas aplicables	Regla a utilizar
carnívoro animal	posee_pelo = si puede_dar_leche = si	3,1,6,7,8,9,10,11	3

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	mamífero= si come_carne = si dientes_puntiagudos = si garras =si ojos_adelante = si		

11ma Ejecución: Como las cuatro premisas de la regla 3 se cumplen, entonces se llega a la conclusión de que se trata de un carnívoro, por lo tanto el objetivo carnívoro pasa a ser parte de los datos conocidos y el objetivo prioritario vuelve a ser animal, siendo la regla 1 la que se debe probar para ver si sus premisas se cumplen.

Objetivos	**Datos conocidos**	**Reglas aplicables**	**Regla a utilizar**
animal	posee_pelo = si puede_dar_leche = si mamífero= si come_carne = si dientes_puntiagudos = si garras =si ojos_adelante = si carnívoro= si	1,6,7,8,9,10,11	1

12ma Ejecución: Se procede a probar la primera premisa de la regla 1 mamífero = si, como es parte de los datos conocidos, se procede a determinar si la segunda premisa carnívoro = si se encuentra en los datos conocidos, como si existe entonces se procede con la tercera premisa color = leonado se encuentra en los datos conocidos, al no existir valor alguno se busca una regla que nos permita derivar dicho valor; como no existe ninguna regla que en su conclusión haga referencia a esta premisa el mecanismo de inferencia procede a preguntar al usuario del sistema ¿Cuál es el color del animal? al usuario. La respuesta afirmativa del usuario confirma la premisa, que posteriormente se incorpora a los datos conocidos.

Objetivos	**Datos conocidos**	**Reglas aplicables**	**Regla a utilizar**
animal	posee_pelo = si puede_dar_leche = si	1,6,7,8,9,10,11	1

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

	mamífero= si come_carne = si dientes_puntiagudos = si garras =si ojos_adelante = si carnívoro= si color=leonado		

13ma Ejecución: Se procede a probar la cuarta premisa de la regla 1 manchas_oscuras = si, primero se busca en los datos conocidos, al no existir valor alguno se busca una regla que nos permita derivar dicho valor; como no existe ninguna regla que en su conclusión haga referencia a esta premisa el mecanismo de inferencia procede a preguntar al usuario del sistema ¿Tiene manchas oscuras? En este caso la respuesta es no, por lo que la premisa no se cumple y falla la regla 1, tomando prioridad la regla 6 que está en la secuencia de las reglas aplicables.

Objetivos	**Datos conocidos**	**Reglas aplicables**	**Regla a utilizar**
animal	posee_pelo = si puede_dar_leche = si mamífero= si come_carne = si dientes_puntiagudos = si garras =si ojos_adelante = si carnívoro= si color=leonado	6,7,8,9,10,11	6

14ta Ejecución: Se procede a probar la primera premisa de la regla 6 mamífero = si, como es parte de los datos conocidos, se procede con la segunda premisa carnívoro = si se encuentra en los datos conocidos como si existe entonces se procede con la tercera premisa color=leonado se encuentra en los datos conocidos como si existe entonces se procede con la cuarta premisa rayas_negras = si al no estar presente en los datos conocidos, se busca una regla que nos permita derivar dicho valor; como no existe

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ninguna regla que en su conclusión haga referencia a esta premisa el mecanismo de inferencia procede a preguntar al usuario del sistema ¿Tiene rayas negras el animal? En este caso la respuesta es si por lo que la premisa se cumple y pasa a ser parte de los datos conocidos. Todas las premisas de la regla 6 se cumplen y por lo tanto el objetivo que era encontrar un animal con ciertas características se cumple, siendo este el tigre.

Objetivos	**Datos conocidos**	**Reglas aplicables**	**Regla a utilizar**
	posee_pelo = si puede_dar_leche = si mamífero= si come_carne = si dientes_puntiagudos = si garras =si ojos_adelante = si carnívoro= si color=leonado rayas_negras = si animal = tigre	7,8,9,10,11	

15ta Ejecución: Las reglas 7,8,9,10,11 son desechadas debido a que ya se encontró con los datos conocidos un animal, que en este caso es el tigre y ya no existe ningún objetivo.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

CAPITULO 3

3 METODOS Y ESTRATEGIAS DE BUSQUEDA

Las estrategias de búsqueda están fundamentadas en la actuación de los agentes que tienen un propósito definido. Estos agentes, ejecutan una serie de acciones que les facilita la transición entre diferentes estados, en la búsqueda de alcanzar el estado objetivo, que representa su meta final. Estas búsquedas se realizan en contextos donde el entorno de trabajo es observable, discreto, estático y determinista. Es importante destacar que la elección de la acción a implementar, con el fin de transitar hacia un nuevo estado, conlleva un costo asociado.

En el proceso de búsqueda se siguen comúnmente 3 pasos que son:

- ✓ **Formulación del objetivo:** Esta fase implica definir el estado deseado al que el agente aspira llegar al concluir el proceso.
- ✓ **Realización de la búsqueda:** En este paso, se identifica la secuencia de estados intermedios, considerando la medida de rendimiento del agente, que guiará la transición desde el estado inicial hasta el estado objetivo.
- ✓ **Ejecución de la búsqueda:** Una vez determinada la ruta óptima, se llevan a cabo las acciones sugeridas durante la fase de realización de la búsqueda para alcanzar el estado final deseado.

3.1 Búsqueda no informada

Esta estrategia de búsqueda opera exclusivamente con la información contenida en el problema propuesto, sin recurrir a datos adicionales. Su objetivo es identificar los estados sucesores y, a través de este proceso, localizar el estado meta. En el contexto de las búsquedas, lo que generalmente se visualiza es un árbol bidimensional. Esto permite realizar búsquedas desplazándose tanto en el eje X como en el eje Y, navegando a través de los nodos del árbol y sus respectivas conexiones.

Las búsquedas no informadas hacer a ser tratadas se indica a continuación:

Búsqueda Preferentemente por Amplitud: Esta técnica explora todos los nodos de un nivel antes de pasar al siguiente.

Búsqueda Primero en Profundidad: Se centra en explorar tan profundo como sea posible a lo largo de una rama antes de retroceder.

Búsqueda de Profundidad Limitada: Similar a la búsqueda en profundidad, pero con una restricción en cuántos niveles se exploran.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Búsqueda por Profundización Iterativa: Combina las ventajas de la búsqueda en profundidad y la búsqueda por amplitud, incrementando gradualmente la profundidad explorada.

Búsqueda de Costo Uniforme: Se enfoca en expandir el nodo con el menor costo acumulado desde el estado inicial.

3.1.1 Búsqueda preferentemente por amplitud

La estrategia de búsqueda preferentemente por amplitud es un método sistemático que comienza desde el nodo raíz y se extiende horizontalmente a través de los nodos adyacentes, abarcando cada nivel de manera exhaustiva antes de descender al siguiente. Esta exploración horizontal garantiza que todos los nodos en un nivel sean visitados y examinados antes de pasar al nivel inferior. Operando bajo el principio FIFO (First In, First Out), esta estrategia asegura que los nodos se expandan en el orden en que fueron descubiertos, lo que facilita un proceso de búsqueda organizado y predecible (Chopra, 2012).

Cada conexión entre los nodos se considera con un costo uniforme, el algoritmo avanza equitativamente en todas las direcciones posibles, un paso a la vez en cada ruta viable. Esta búsqueda es particularmente eficaz en escenarios en los cuales todas las rutas deben ser evaluadas por igual o cuando la estructura del problema no proporciona pistas claras sobre cuál podría ser la ruta más prometedora.

Aunque esta estrategia garantiza encontrar la solución óptima si existe dentro del espacio de búsqueda, su exhaustividad puede resultar en un consumo de tiempo y recursos mayor al mínimo necesario. Es especialmente efectiva en árboles de búsqueda con una profundidad y número de ramas moderados, pero puede volverse menos práctica en árboles con una gran cantidad de nodos o niveles, debido a que requiere almacenar y procesar todos los nodos de un nivel antes de avanzar al siguiente.

La búsqueda por amplitud se aplica en diversos contextos de la inteligencia artificial y la informática, como en la solución de puzzles, navegación en mapas, y en la planificación de rutas. Su capacidad para dar una solución óptima y su naturaleza sistemática la hacen una herramienta valiosa en problemas donde la precisión y la exhaustividad son más críticas que la eficiencia en términos de tiempo o memoria (Russell & Norvig, 2012).

Como *ejemplo*; se tiene un árbol binario (figura 3.1) donde el estado inicial es el

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

nodo A y el objetivo es poder llegar al **nodo I** utilizando la búsqueda preferentemente por amplitud y pasando a través de los estados intermedios que son los nodos que forman el camino hacia la meta.

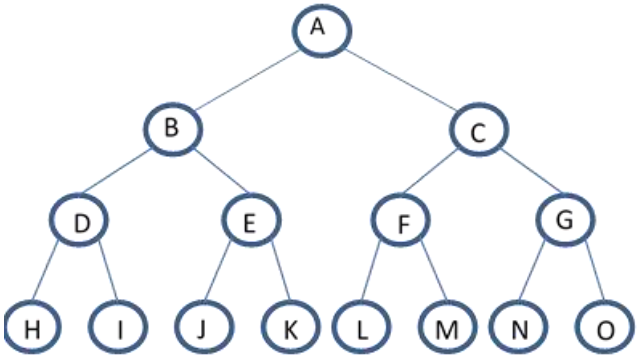
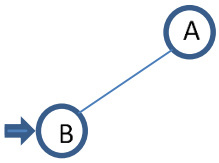


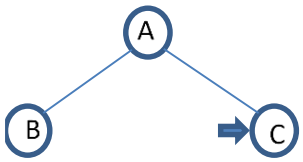
Figura 3.1 Árbol binario

Elaborado por los Autores

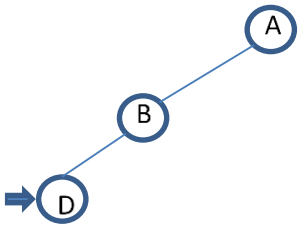
Se empieza en el nivel 0 donde se encuentra el nodo raíz que para este caso es el nodo A, como no coincide este nodo con el objetivo que es el nodo I se pasa al siguiente nivel (nivel 1).



Se pasa al nodo que está a la izquierda del árbol binario que en este caso es el nodo B, pero al no coincidir con el nodo objetivo, se procede a pasar al nodo siguiente en horizontal.

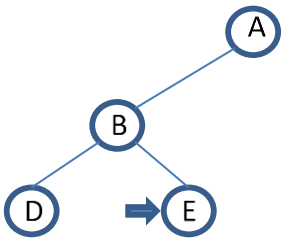


El siguiente nodo en ser visitado es el nodo C, que tampoco constituye el objetivo (nodo I), por lo cual se desciende al siguiente nivel (nivel 2).

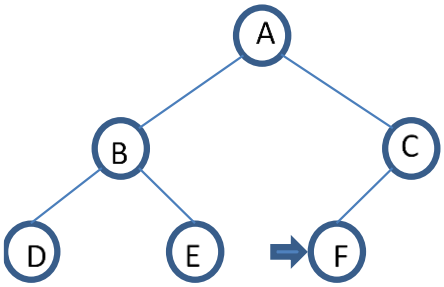


En el segundo nivel el primer nodo en ser visitado es el nodo D al no ser el nodo objetivo es desechado.

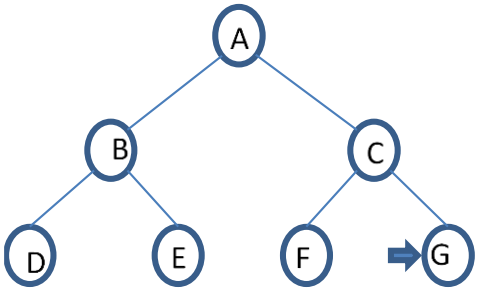
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



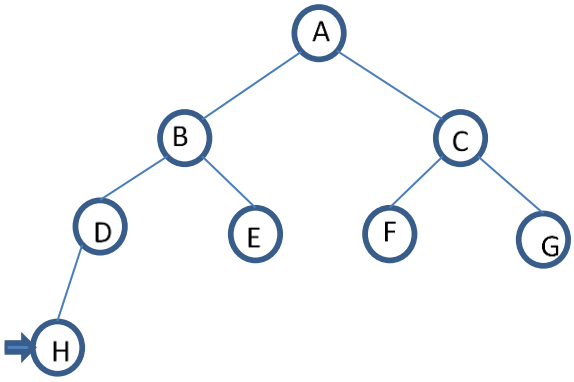
A continuación, se debe trasladar al nodo E que se encuentra en el segundo nivel, que no constituye el nodo objetivo. Por lo cual se deben visitar los nodos hijos del nodo C que están en el segundo nivel.



El nodo F es el siguiente en ser visitado en el proceso, pero sucede lo mismo que en los casos anteriores, no coincide con el nodo objetivo.

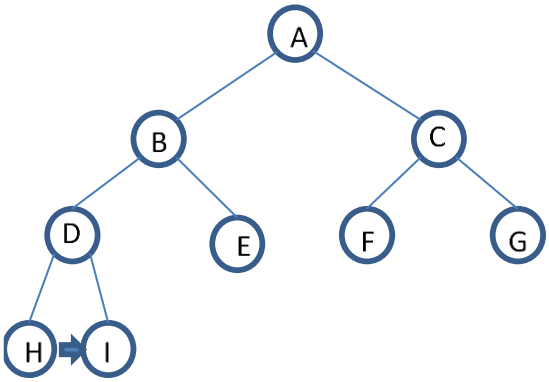


El nodo G es el último en ser visitado en el nivel 2 y tampoco concuerda con el nodo objetivo que es el nodo I, por lo cual se debe descender al nivel 3 en búsqueda del objetivo.

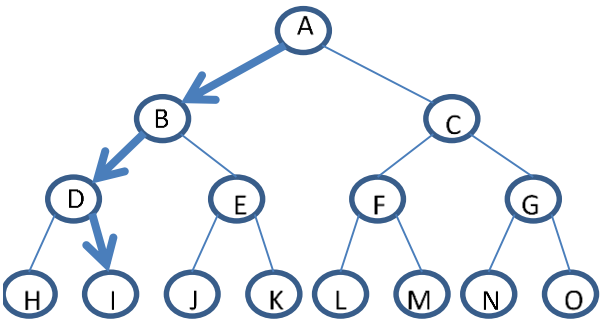


El primer nodo a ser visitado en el nivel 3 es el nodo H al no ser el objetivo, continua la búsqueda hacia el nodo vecino en el eje de las X, cuyo padre es el nodo D.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



En el nivel 3 se encuentra el nodo I que era el objetivo, por lo cual se ha llegado al estado final.



La solución obtenida utilizando la búsqueda por amplitud para ir del nodo A al nodo I constituye el camino a través de los nodos A-B-D-I.

3.1.2 Búsqueda primero en profundidad

La estrategia de búsqueda primero en profundidad se caracteriza por su enfoque vertical, procediendo a lo largo del eje Y. Inicia desde el nodo raíz y se dirige hacia el nodo sucesor más a la izquierda en el siguiente nivel, y continúa descendiendo por el árbol siguiendo siempre la rama más a la izquierda. Este proceso se repite hasta alcanzar un punto donde no existen más nodos sucesores. Si no se encuentra el nodo objetivo, el algoritmo retrocede al nodo anterior para explorar otros sucesores posibles, siguiendo el principio LIFO (Last In, First Out) (Russell & Norvig, 2012). Esta metodología permite que la búsqueda se sumerja profundamente en una dirección antes de considerar otras alternativas, lo que puede ser especialmente eficiente en situaciones donde la solución se encuentra en niveles más profundos del árbol.

En el mejor de los escenarios, si el algoritmo 'tiene suerte' y elige consistentemente el camino correcto hacia el nodo objetivo, la búsqueda en profundidad puede ser extremadamente rápida y directa. Sin embargo, a diferencia de la búsqueda por amplitud, no garantiza encontrar la solución óptima si existe más de una, ya que el primer camino exitoso que encuentra será el seleccionado.

Por otro lado, en el peor de los casos, este algoritmo podría explorar cada posible

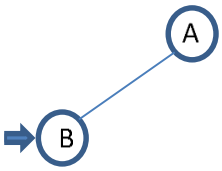
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

camino en el árbol de búsqueda antes de encontrar la solución, lo que resultaría en un tiempo de ejecución significativamente más largo, especialmente en árboles con muchas ramificaciones y profundidades. Aunque este enfoque es menos eficiente en términos de memoria comparado con la búsqueda por amplitud, su simplicidad lo hace valioso en ciertos contextos, como en la solución de laberintos o problemas que requieren explorar todas las posibilidades hasta encontrar una solución viable.

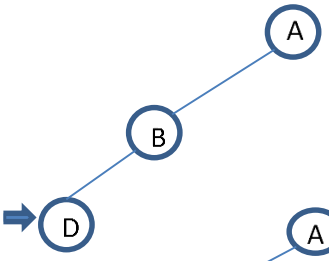
Es importante destacar que, debido a su naturaleza, la búsqueda en profundidad es más susceptible a quedar atrapada en ciclos o caminos sin salida, por lo que a menudo se implementan mecanismos como la limitación de profundidad o la verificación de estados repetidos para mejorar su eficacia y evitar la exploración innecesaria de caminos ya evaluados.

Utilizando el mismo árbol empleado en la búsqueda preferentemente por amplitud y siendo el nodo objetivo el nodo I se tiene:

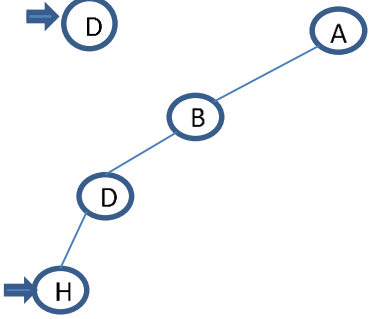
El primer nodo en ser visitado es el nodo A, al no ser el nodo objetivo se desciende al siguiente nivel (nivel 1)



En el nivel 1 se encuentra el nodo B que no es el objetivo y se procede a bajar al nivel 2.

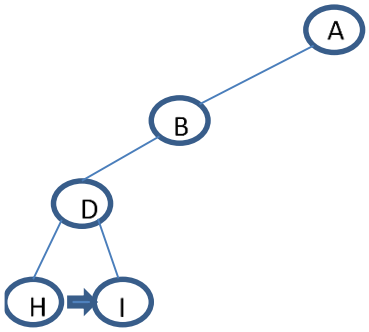


El nodo D que se encuentra en el nivel 2 tampoco es el objetivo por lo cual se procede a bajar al nivel 3.



Al no ser el nodo H el nodo objetivo y no existir más sucesores se retrocede al nodo D para ver si este tiene otros sucesores, en este caso si tiene otros sucesores.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



El nodo D tiene un sucesor que es el nodo I, que constituye el objetivo. Por lo que se llega a la solución que es ir desde el nodo A pasando por los nodos B y D hasta llegar al nodo objetivo que es el nodo I (A-B-D-I).

3.1.3 Búsqueda de profundidad limitada

En el contexto de las estrategias de búsqueda aplicadas a estructuras de árbol, la técnica de búsqueda primero en profundidad es una de las más utilizadas, caracterizada por explorar lo más profundo posible a lo largo de una rama antes de considerar otras alternativas. Sin embargo, en árboles con una gran cantidad de niveles, esta técnica puede ser ineficiente, debido a que podría implicar recorrer un número extenso de nodos antes de encontrar la solución.

Para optimizar este tipo de búsqueda en árboles de gran profundidad y mitigar la posibilidad de exploraciones excesivamente largas, se introduce el concepto de "Búsqueda de Profundidad Limitada". Esta variante de la búsqueda impone un límite de profundidad, estableciendo un nivel máximo al cual se realizará la exploración. Este límite es particularmente importante cuando se tiene información o estimaciones previas las cuales sugieren que el nodo objetivo está ubicado dentro de un rango de profundidad específico (Russell & Norvig, 2012).

Al alcanzar el límite de profundidad preestablecido, la búsqueda trabaja bajo la premisa de que no existen sucesores adicionales en ese nivel, aunque esto puede ser que no sea cierto en la estructura original del árbol. Esta suposición permite delimitar la exploración, enfocando los recursos y esfuerzos en una sección más manejable del árbol, lo que resulta en una mayor eficiencia y rapidez en encontrar una solución.

Esta técnica acelera significativamente la búsqueda en ciertos casos, pero también presenta el riesgo de omitir la solución si el nodo objetivo se encuentra más allá del límite de profundidad impuesto. La elección de un límite de profundidad adecuado es un factor crítico, que se debe balancear entre la eficiencia deseada y la probabilidad de encontrar el nodo objetivo.

La búsqueda de profundidad limitada se utiliza en problemas donde la meta se

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

espera que se encuentre a una profundidad moderada, o en situaciones en las cuales es preferible obtener una respuesta rápida, aunque sea parcial, en lugar de realizar una exploración completa. Esta técnica es un ejemplo de cómo las estrategias de búsqueda se adaptadan y ajustadan para satisfacer las necesidades específicas de diferentes problemas y contextos.

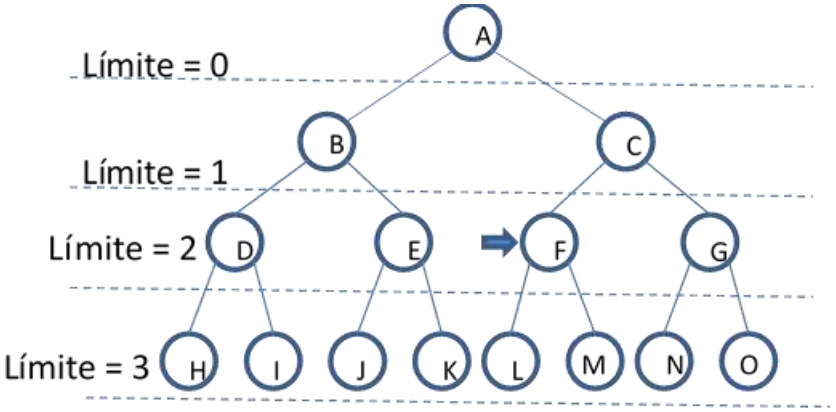


Figura 3.2 Árbol de búsqueda

Elaborado por los Autores

Para el ejemplo que estamos utilizando si el objetivo es llegar al nodo F (figura 3.2) si se utiliza la búsqueda primero en profundidad se necesita pasar por diez nodos para llegar a la solución. Pero se va a ver lo que sucede si se utiliza un límite = 2 (figura 3.3), en el cual se tiene una especie de árbol truncado.

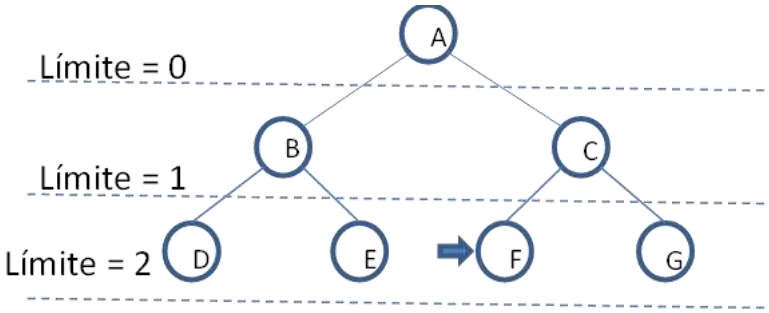
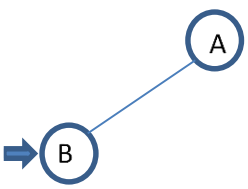


Figura 3.3 Árbol con límite de 2

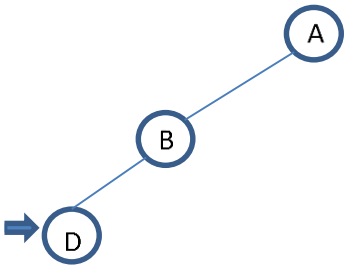
Elaborado por los Autores

El primer nodo en ser visitado es el nodo A al no ser el nodo objetivo se desciende al siguiente nivel (nivel 1)

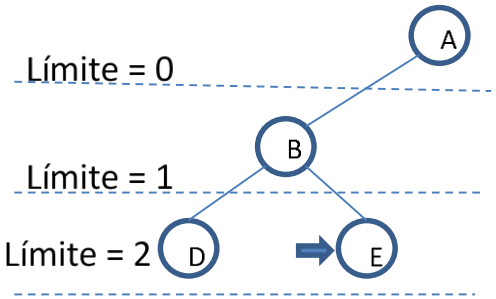


En el nivel 1 se encuentra el nodo B que no es el nodo objetivo y se procede a bajar al nivel 2.

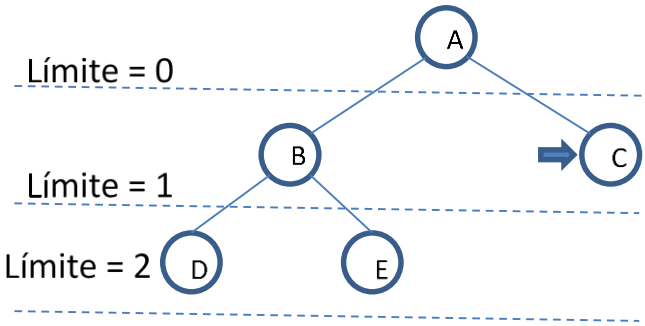
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



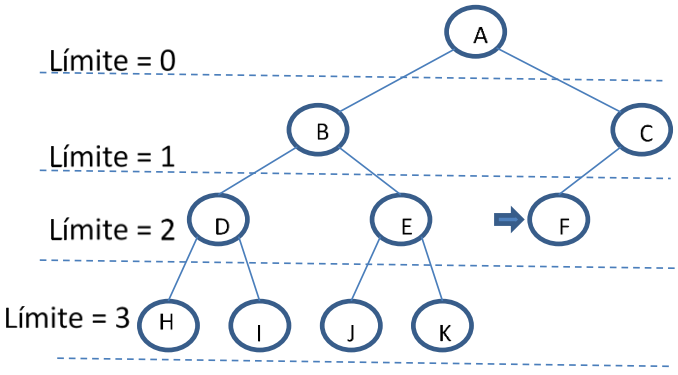
El nodo D que se encuentra en el nivel 2 tampoco es el objetivo por lo cual se procede a retroceder al nodo B en vista de que se ha llegado al límite = 2 y por tanto se considera que el nodo D ya no tiene más sucesores.



Se retrocede al nodo B y se tiene otro sucesor que es el nodo E, este nodo es visitado, pero al no ser la solución esto es el objetivo nodo F, se procede a retroceder al nodo del que desciende esto es al nodo B, para observar si tiene otros hijos, pero al no ser el caso se retrocede aún más hasta llegar al nodo A.



El nodo A tiene además del nodo B otro hijo que es el nodo C, el cual no es la solución por lo cual se procede a descender hacia el nodo sucesor.



El nodo F que constituye el sucesor del nodo C es el nodo objetivo, por lo cual se detiene la búsqueda y se encuentra la solución, que es ir del nodo A al nodo C y de ahí al nodo objetivo que es el nodo F (A-C-B). Como se puede observar nunca se utilizó el límite=3 para encontrar el nodo objetivo.

3.1.4 Búsqueda por profundización iterativa

La búsqueda por profundización iterativa es una estrategia innovadora que integra

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

los aspectos más eficientes de la búsqueda primero en profundidad y la búsqueda de profundidad limitada. Esta técnica se basa en la realización de una serie de búsquedas en profundidad, cada una con un límite de profundidad que se incrementa de manera iterativa en cada ciclo.

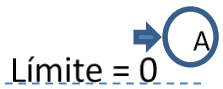
El proceso comienza con un límite de profundidad inicial fijado en 0, explorando solo el nodo raíz. Si el nodo objetivo no se encuentra en este nivel superficial, el límite se aumenta en una unidad y se procede con una nueva búsqueda en profundidad, esta vez extendiéndose hasta el siguiente nivel. Este enfoque iterativo se repite, incrementando el límite en cada ciclo, hasta encontrar el nodo objetivo o haber recorrido completamente la estructura del árbol.

Una ventaja significativa de la búsqueda por profundización iterativa es su eficiencia al explorar árboles de búsqueda extensos. Esta metodología es especialmente útil cuando se desconoce la profundidad exacta del nodo objetivo. En lugar de comprometerse a una búsqueda de toda la profundidad del árbol desde el comienzo, que puede ser costosa en términos de tiempo y memoria, esta estrategia adopta un enfoque más controlado y escalonado, examinando cada nivel de manera secuencial.

Este método combina la eficiencia de la búsqueda primero en profundidad con respecto al uso de memoria, con la garantía de la búsqueda por amplitud de encontrar la solución óptima sin incurrir en altos costos asociados con la memoria. Es efectivo en situaciones donde el árbol de búsqueda es muy amplio o en situaciones en las cuales la ubicación del nodo objetivo es incierta; garantiza una exploración sistemática y minuciosa, sin la sobrecarga de revisar innecesariamente nodos profundos desde el inicio.

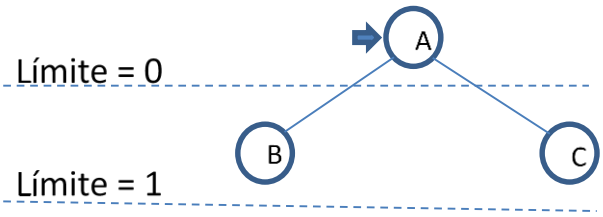
Por lo tanto, la búsqueda por profundización iterativa se posiciona como una herramienta poderosa en la resolución de problemas de inteligencia artificial, ofreciendo un equilibrio óptimo entre eficiencia y exhaustividad, adecuado para una amplia gama de aplicaciones donde la estructura del problema es compleja y la ubicación del objetivo no es predecible.

Para ejemplo que se utiliza el proceso sería el siguiente:

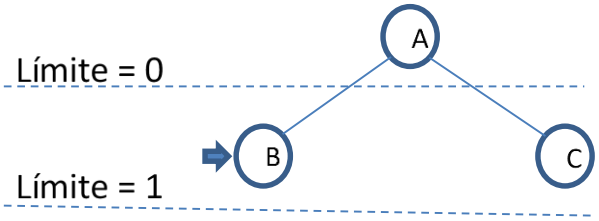


En el límite = 0 se tiene un solo nodo sin sucesores. El objetivo es el nodo F, como el nodo raíz es el nodo A y no constituye el nodo objetivo se procede a aumentar el límite en 1 (0 +1).

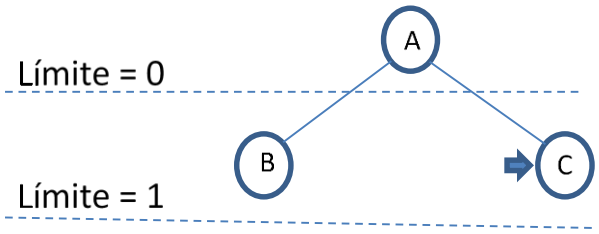
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



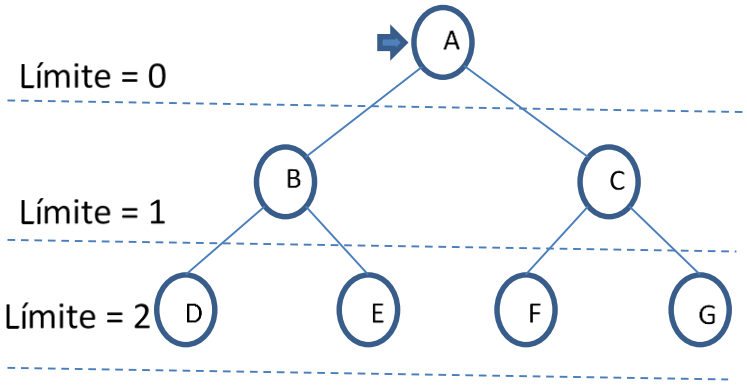
El nuevo árbol de búsqueda tiene 3 nodos y empieza en el nodo A que al no ser el nodo objetivo se procede a bajar al siguiente nivel, por el enlace que está más a la izquierda.



En el nivel 1 se encuentra el nodo B que no es el nodo objetivo por lo que se regresa al nodo padre que es el nodo A.

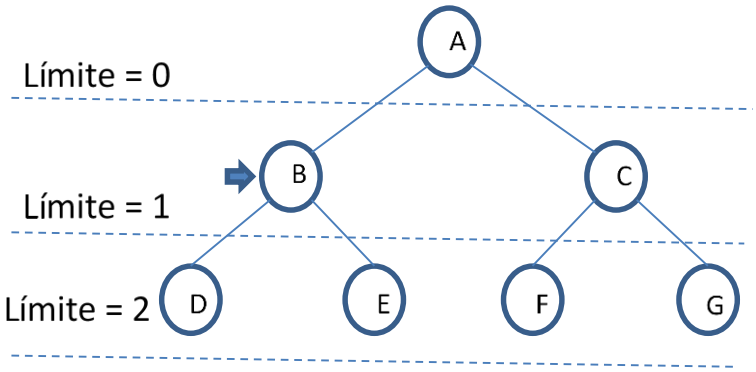


El nodo siguiente a visitar es el nodo C, que es el sucesor del nodo A pero no constituye el nodo objetivo, por lo tanto se regresa al nodo padre que es el nodo A y al no tener más sucesores, se procede a aumentar el límite = 2 (1+1).

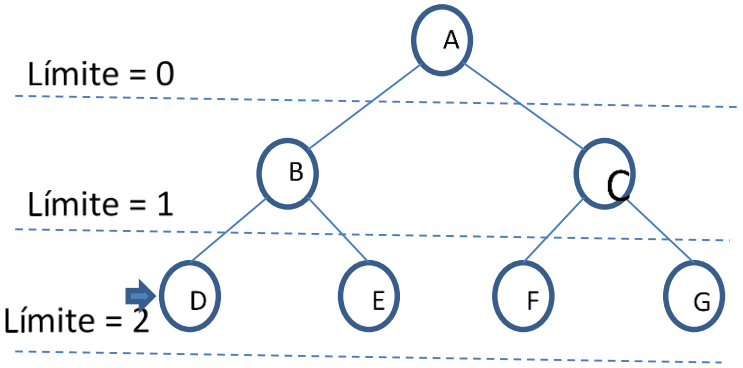


El nuevo árbol de búsqueda muestra los niveles correspondientes, en el nivel = 0 el nodo A raíz no es el nodo objetivo por lo cual se procede a descender hacia uno de sus sucesores.

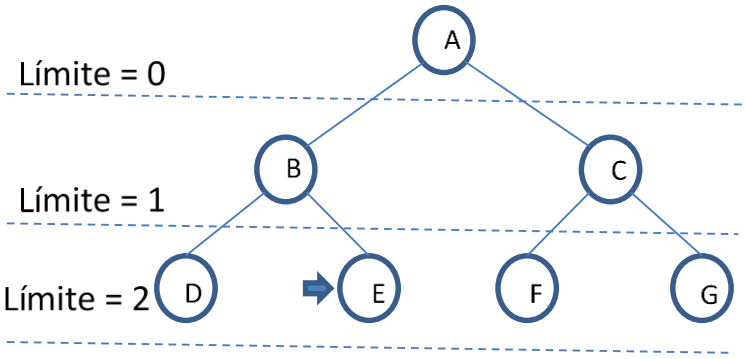
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



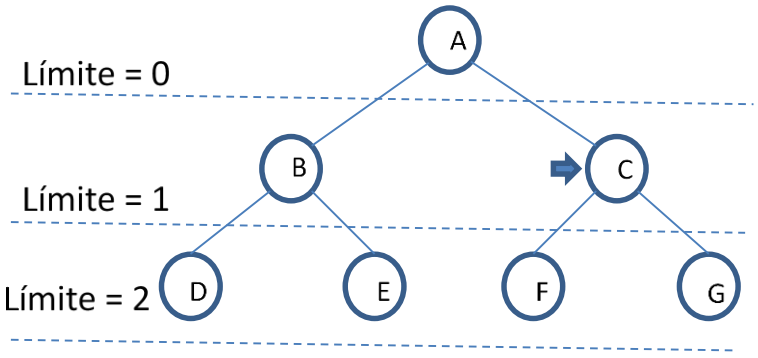
El nodo B no constituye el nodo objetivo por lo que se procede a descender en búsqueda del nodo hijo.



El nodo D no es el nodo objetivo por lo que se regresa al padre que es el nodo B para ir al siguiente sucesor.



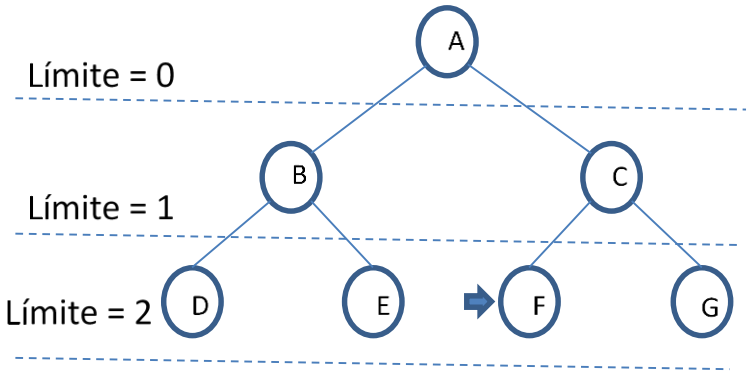
El nodo E es el otro hijo del nodo B pero no constituye el nodo objetivo, al no tener sucesores el nodo E se regresa al nodo padre. El nodo B no tiene más hijos por lo cual se retrocede hacia el padre del nodo B regresando al nodo raíz nodo A.



El nodo A tiene además de B otro sucesor que es el nodo C pero este nodo tampoco coincide con el nodo objetivo; por lo cual se procede a descender hacia uno de sus sucesores, en este caso siempre el que está

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

más a la izquierda.



El nodo F es el nodo sucesor del nodo C y constituye el nodo objetivo por lo cual la búsqueda termina y se tiene la solución que constituye ir del nodo A al nodo C y de ahí al nodo F.

3.1.5 Búsqueda de costo uniforme

Esta estrategia de búsqueda se diferencia de otras técnicas tradicionales en que no se basa únicamente en la cantidad de pasos o nodos que se deben atravesar para llegar al objetivo, sino en el costo acumulado asociado a esos pasos. Es especialmente relevante en escenarios donde los enlaces entre nodos tienen costos variables y no unitarios.

En la búsqueda de costo uniforme, la prioridad no es la proximidad superficial al objetivo (como podría ser el caso en una búsqueda primero en anchura), sino el camino que representa el menor costo acumulado (Russell & Norvig, 2012).

En cada etapa de la búsqueda, se expande el nodo que, hasta ese momento, tiene el camino de menor costo desde el nodo inicial.

Aunque a simple vista podría parecer similar a la búsqueda primero en anchura, la búsqueda de costo uniforme mejora y refina este enfoque al considerar el costo de los enlaces en su proceso de decisión.

No se trata simplemente de contar los pasos o nodos atravesados, sino de evaluar y sumar los costos asociados a esos pasos. De esta manera, la estrategia garantiza que, al alcanzar el nodo objetivo, se haya seguido el camino con el costo total más bajo, independientemente del número de nodos que se haya tenido que atravesar.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

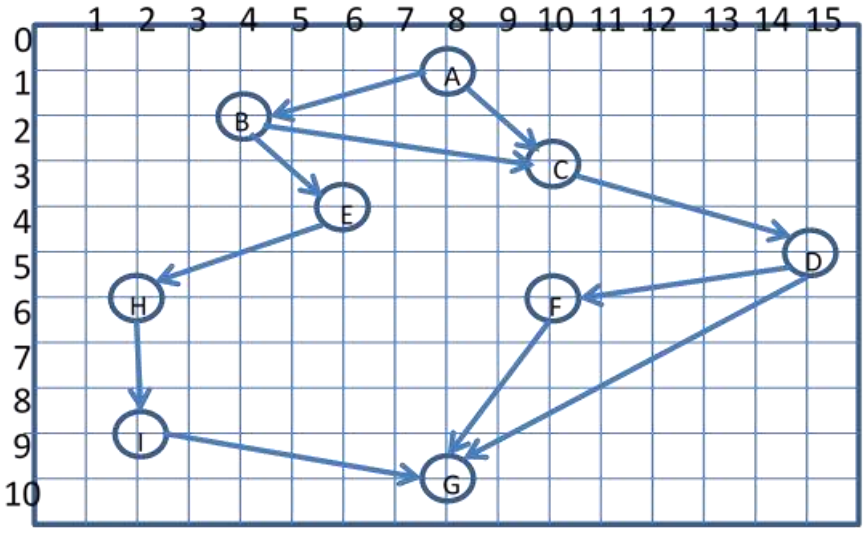


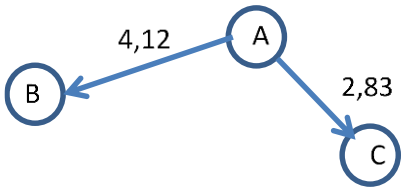
Figura 3.4 Grafo de búsqueda

Elaborado por los Autores

Esta técnica es esencial en problemas donde las decisiones no solo dependen de la distancia o cantidad de pasos, sino también de factores como tiempo, recursos o cualquier otra métrica que pueda ser cuantificada en términos de costo.

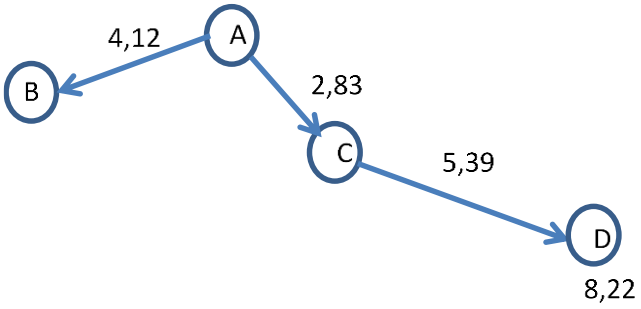
Se va a utilizar el siguiente grafo (figura 3.4) para realizar la búsqueda desde el nodo origen que en este caso será el nodo A hasta el nodo destino que será el nodo G. Las distancias entre los nodos se obtendrán mediante el trazo de líneas rectas entre los centros de los nodos y se respetarán las direcciones señaladas mediante las flechas

El costo del nodo A es cero, y no constituye el nodo destino, por lo cual se procede a determinar el costo para llegar hacia sus sucesores.

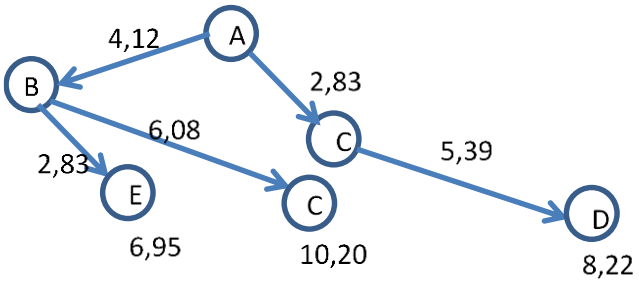


El costo de ir del nodo A hacia el nodo B se obtiene aplicando el teorema de Pitágoras $\sqrt{4^2 + 1^2} = 4{,}12$ lo mismo se aplica para ir al nodo C $\sqrt{2^2 + 2^2} = 2{,}83$

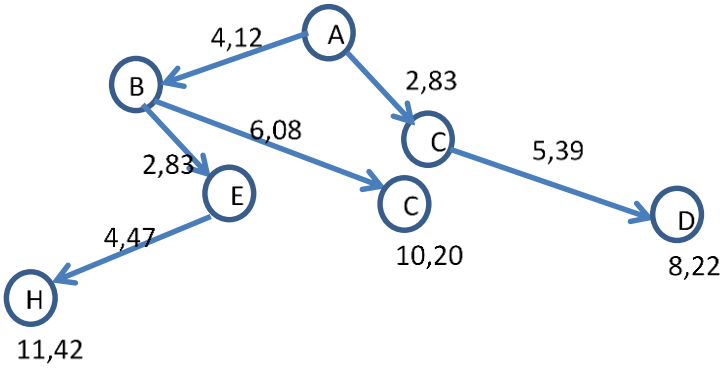
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



El costo de ir del nodo C al nodo D es $\sqrt{5^2 + 2^2} = 5{,}39$. Como se suman los costos de ir del nodo A hasta el nodo D el valor es 2,83 +5,39 =8,22. Como C no tiene más descendientes se procede a comparar en el árbol que se ha formado los costos entre el nodo B y nodo D que son los extremos del árbol y no tienen más sucesores, por lo que el nodo de menor costo es B 4,12.

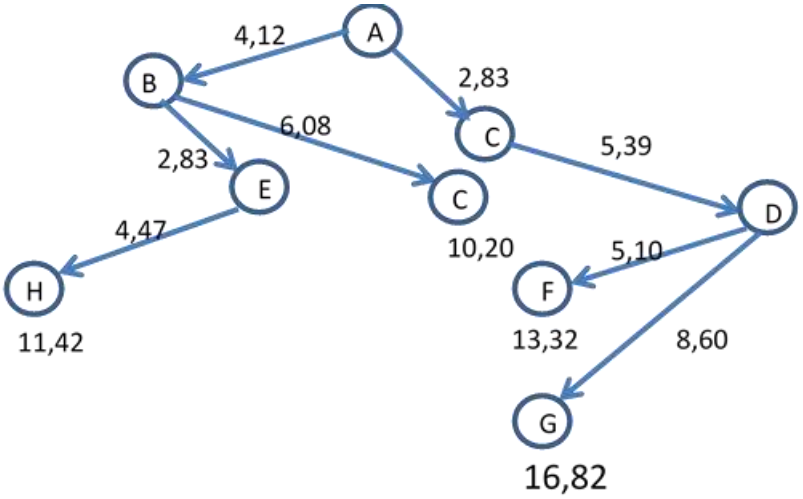


Los dos sucesores del nodo B son el nodo E y el nodo C. El costo para ir del nodo B al nodo E es $\sqrt{2^2 + 2^2} = 2{,}83$ y el costo para ir del nodo B al nodo C es $\sqrt{6^2 + 1^2} = 6{,}08$. Se comparan los totales entre los nodos que no tienen sucesores en este nuevo árbol que son el E, C y D; siendo el de menor valor el E, por lo cual se procede a expandir este nodo.



El nodo E tiene un solo sucesor que es el nodo H y el costo de ir del nodo E al nodo H es $\sqrt{4^2 + 2^2} = 4{,}47$. El total de la suma de ir del nodo A al nodo H es 11,42 comparando con los valores para llegar al nodo C y al nodo D; el nodo D tiene un menor costo por lo cual se procede a expandir.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



El nodo D tiene dos sucesores que son el nodo F y el nodo G, el costo de ir del nodo D al nodo F es $\sqrt{5^2 + 1^2} = 5{,}10$ y el costo de ir del nodo D al nodo G es $\sqrt{7^2 + 5^2} = 8{,}60$. El total para ir del nodo A hasta el nodo F es 13,32 y el total para ir del nodo A al nodo G es 16,82. En este caso hemos llegado al nodo objetivo que es el nodo G pero el total es superior a otros nodos como H, C y F que pueden ser expandidos y también llegan al nodo G. Si el objetivo es simplemente llegar al nodo G se habría cumplido con el propósito y el camino sería ir del nodo A a través de los nodos intermedios C y D para llegar al nodo objetivo que es el nodo G. Pero si el propósito es encontrar el camino de menor costo se debería continuar con la búsqueda porque aún existen otros nodos que le lleva al objetivo que es el nodo G. En el caso del ejemplo se debería continuar la búsqueda por los otros caminos que faltan probar que son: A-B-E-H-I-G; A-B-C-D-F-G; A-B-C-D-G, para ver cuál es el camino que tiene el menor costo total comparado con el camino encontrado que es A-C-D-G para determinar si este camino constituye el de menor costo.

3.2 Búsqueda informada

La búsqueda informada, a diferencia de las técnicas de búsqueda no informada, incorpora información adicional que va más allá de la mera descripción del problema. Esta información adicional, denominada heurística, se basa en el conocimiento previo o experiencia relacionada con el problema en cuestión. La heurística no garantiza una solución correcta, pero proporciona una guía o dirección que puede hacer que la búsqueda sea más eficiente (Edelkamp & Schrödl, 2011).

El conocimiento heurístico actúa como una especie de "intuición" que permite estimar la proximidad o calidad de un estado particular respecto al objetivo deseado. Por ejemplo, en un problema de ruta más corta, una heurística podría ser la distancia en línea recta desde un punto a otro, que, aunque no considera obstáculos o caminos específicos, proporciona una estimación rápida de la cercanía al objetivo.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

En la búsqueda informada, se utilizan diversas métricas para evaluar y priorizar los nodos en el espacio de búsqueda. Estas métricas, basadas en heurísticas, permiten ordenar o clasificar los nodos según su potencial o probabilidad de acercarse a la solución óptima. De esta manera, la búsqueda se orienta hacia las opciones más prometedoras, reduciendo el tiempo y los recursos necesarios para encontrar una solución.

Se tratarán dos formas de búsqueda informada:

- ✓ Búsqueda avara primero el mejor
- ✓ Búsqueda A*

3.2.1 Búsqueda avara (primero el mejor)

La estrategia de búsqueda avara, también conocida como "Primero el Mejor", se centra en la exploración prioritaria de aquellos nodos que, según una función heurística, parecen ser los más prometedores para alcanzar el objetivo de manera eficiente. Esta función heurística, representada por $h(n)$, estima el costo desde el nodo actual n hasta el nodo objetivo, basándose en conocimientos previos o heurísticos del problema en cuestión (Kumar, 2013).

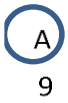
En cada paso de la búsqueda, se selecciona y expande el nodo cuyo valor de $h(n)$ es el menor, es decir, aquel que se estima está más cerca del objetivo. Esta elección se realiza con la esperanza y la premisa de que dirigirse hacia el nodo más prometedor acelerará el proceso de encontrar la solución.

Es importante señalar que, una vez que se alcanza el nodo objetivo, el valor de h(n) se convierte en cero, indicando que no hay más costo estimado pendiente desde ese punto.

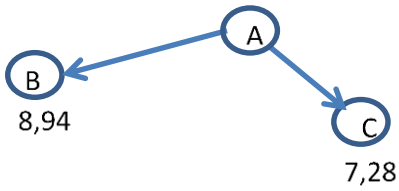
La efectividad de la búsqueda avara está intrínsecamente ligada a la precisión y calidad de la función heurística empleada. Una función heurística bien diseñada puede guiar al algoritmo directamente hacia la solución, mientras que una función imprecisa podría desviar la búsqueda, aumentando el tiempo y los recursos necesarios para encontrar la solución.

Se va a utilizar para ejemplificar esta búsqueda el grafo de búsqueda de costo uniforme, con la métrica de que la distancia más corta entre dos puntos es la línea recta.

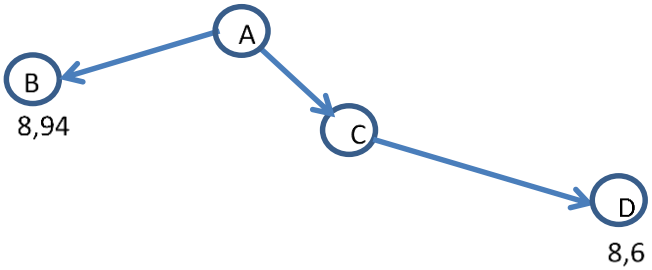
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



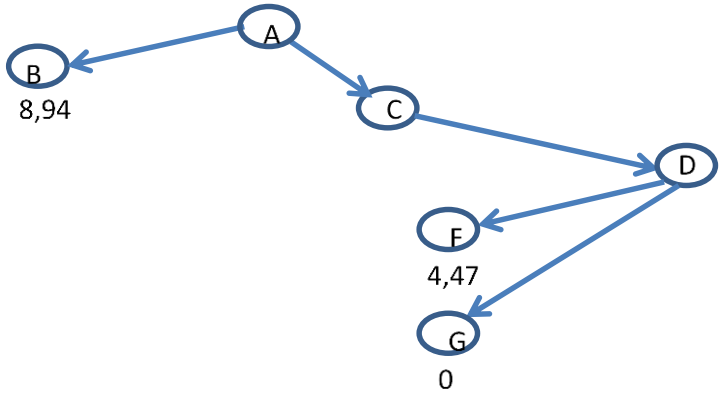
El estado inicial es A y la distancia en línea recta al objetivo G es h(A) = 9 , observe que el nodo A esta en la posición (8,1) y el nodo G en la posición (8,10) entonces la distancia en línea recta es (10 – 1) = 9 por tanto f(A) = 9, para llegar al nodo objetivo se requiere que la heurística sea cero, por lo que no ha llegado al objetivo y se debe expandir el nodo A.



Se procede a expandir el nodo A y se calcula la distancia en línea recta desde los nodos B y C al nodo objetivo G, los valores son: h(B) = $\sqrt{(8-4)^2+(10-2)^2} = 8{,}94$ y h(C) = $\sqrt{(10-8)^2+(10-3)^2} = 7{,}28$, siendo las funciones f(B)= 8,94 y f(C)=7,28.



Se expande el nodo C que tiene menor valor el cual posee solo un nodo sucesor h(D) = $\sqrt{(15-8)^2+(10-5)^2} = 8{,}6$ el valor de la función f(D) = 8,6.



Se comparan los valores de los nodos B y D que no tienen sucesores en el gráfico anterior para ver cuál de los dos se expande, se procede con el nodo D que tiene el menor valor, siendo sus sucesores los nodos F y G (Sipser, 1996). Los valores son h(F) = $\sqrt{(10-8)^2+(10-6)^2} = 4{,}47$ y h(G) = $\sqrt{(0)^2+(0)^2} = 0$, por tanto los valores de f(F) = 4,47 y de f(G)= 0 siendo el final de la búsqueda debido a que se ha alcanzado el nodo objetivo. El camino para ir del nodo A al nodo G es entonces: A - C - D - G.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

3.2.2 Búsqueda A*

Esta búsqueda es conocida como A estrella la cual evalúa los nodos bajo la fórmula f(n) = g(n) + h(n), donde g(n) se utiliza para evaluar el costo de ir por el camino de un nodo hacia su sucesor y h(n) para ir del sucesor hacia el nodo objetivo utilizando la heurística, la suma da el valor de la función f(n) que constituye el costo más barato de ir hacia el objetivo. Al combinar estos dos valores, el algoritmo tiene una forma más precisa de determinar el costo de la solución y optimizar sus opciones sobre la marcha. El algoritmo realiza un seguimiento de costo (costo de la ruta hasta ahora + costo estimado hasta el objetivo), y una vez que excede el costo estimado de alguna opción anterior, el algoritmo abandonará la ruta actual y volverá a la opción anterior, evitando así ir por un camino largo e ineficiente que h(n) marcó erróneamente como el mejor. Este algoritmo al basarse en una heurística, es tan bueno como la heurística que emplea, para que la búsqueda A* sea óptima, la función heurística, h(n) para cada nodo n y nodo sucesor n' con costo de paso c, $h(n) \leq h(n') + c$ (Russell & Norvig, 2012).

El algoritmo de búsqueda A* es una de las estrategias más reconocidas y efectivas en el ámbito de la inteligencia artificial para encontrar el camino más corto en un espacio de búsqueda. Su eficacia radica en su capacidad para combinar información sobre los costos ya conocidos y las estimaciones heurísticas de los costos futuros.

La fórmula que guía esta búsqueda es f(n)=g(n)+h(n):

g(n) representa el costo real desde el nodo de inicio hasta el nodo actual n.

h(n) es una función heurística que estima el costo desde el nodo n hasta el nodo objetivo.

La suma de estos dos valores, f(n), proporciona una estimación del costo total desde el nodo de inicio hasta el objetivo pasando por el nodo.

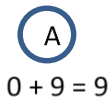
La magia del algoritmo A* radica en cómo utiliza esta información. En cada paso, el algoritmo expande el nodo con el valor f(n) más bajo, asegurando que siempre esté explorando el camino más prometedor hacia el objetivo. Si, en algún momento, el costo acumulado de un camino supera una estimación previa, A* descarta ese camino, evitando así exploraciones innecesarias y optimizando el proceso de búsqueda.

Para que A* garantice encontrar la solución más óptima, es importante que la función heurística h(n) no sobreestime el costo real para llegar al objetivo.

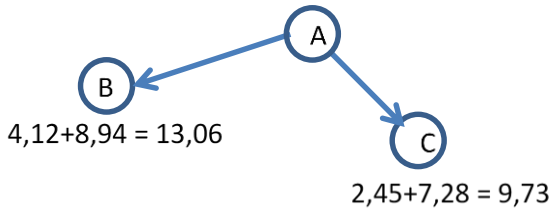
ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Matemáticamente, esto se expresa como: para cada nodo n y su sucesor n' con un costo de paso c, debe cumplirse que h(n)≤h(n')+c.

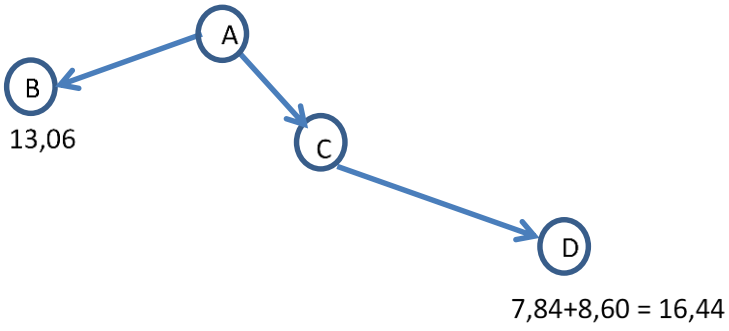
En esencia, A* es un algoritmo que combina lo mejor de las búsquedas informadas y no informadas, proporcionando soluciones eficientes y precisas, siempre y cuando se cuente con una heurística adecuada.



Para la demostración se utilizará el grafo de la búsqueda de costo uniforme, el estado inicial es el nodo A y el nodo objetivo es el nodo G. el valor de g(A) es 0 porque no existe ningún desplazamiento, mientras que el valor de h(A) es (10 -1), por las posiciones que mantiene los nodos igual que en la búsqueda avara primero el mejor. Entonces f(A) = g(A) +h(A)=0 + 9 = 9.



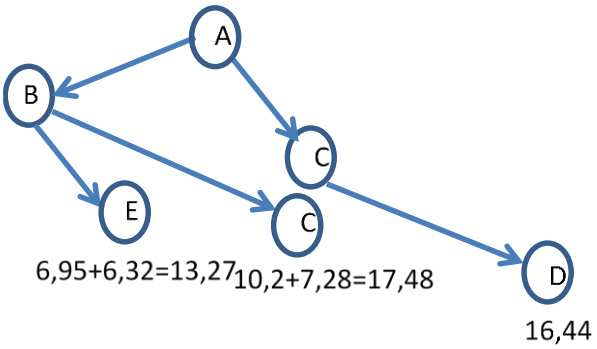
Se procede a expandir los nodos hijos del nodo A que son los nodos B y C, se calcula el valor de f(B)= g(B) + h(B) = $\sqrt{(8-4)^2+(2-1)^2}+\sqrt{(8-4)^2+(10-2)^2}$ = 4,12 + 8,94 = 13,06, así mismo se procede a calcular el valor de f(C) = $\sqrt{(10-8)^2+(3-1)^2}+\sqrt{(10-8)^2+(10-3)^2}$ = 2,45 + 7,28 = 9,73. El nodo que tiene el menor valor es C el cual se procede a expandir.



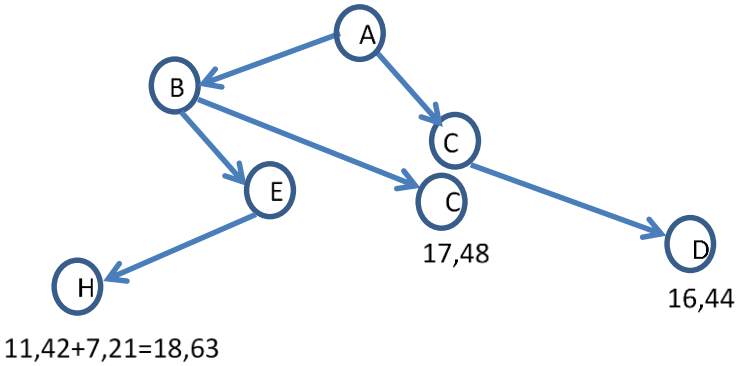
El nodo C tiene un solo sucesor que es el nodo D, se procede a calcula el valor de g(D) que es la distancia de ir del nodo A al nodo C más la distancia de ir del nodo C al nodo D, entonces g(D) = 2,45 + $\sqrt{(15-10)^2+(5-3)^2}$ = 2,45 + 5,39 = 7,84 ; el valor de h(D) = $\sqrt{(15-8)^2+(10-5)^2}$ = 8,60. El valor de f(D) = 7,84 + 8,60 = 16,44. El nodo que se procede a expandir es el que tiene el menor valor que en este caso es el nodo

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

B.

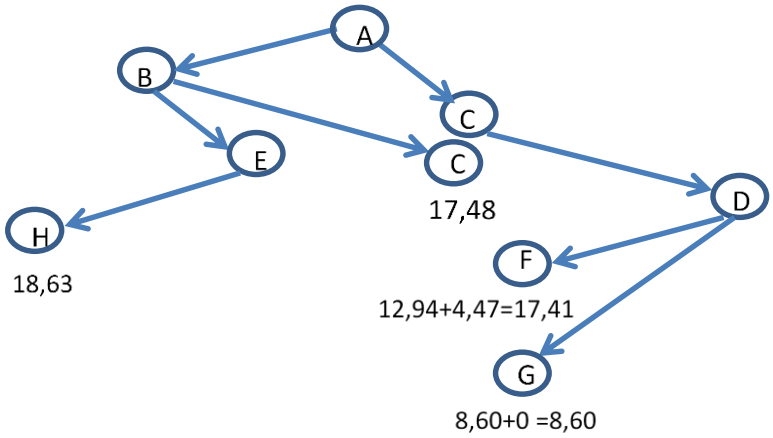


El nodo B tiene 2 sucesores que son el nodo E y el nodo C, se procede a realizar los cálculos para cada uno de estos nodos. El valor de g(E) = 4,12 + $\sqrt{(6-4)^2+(4-2)^2}$ = 4,12 + 2,83 = 6,95 ; el cálculo de la heurística es h(E) = $\sqrt{(8-6)^2+(10-4)^2}$ = 6,32 ; el valor de f(E) = 6,95 + 6,32 = 13,27 . El valor de g(C)= 4,12 + $\sqrt{(10-4)^2+(3-2)^2}$ = 4,12 + 6,08 = 10,2 ; se procede a calcular el valor de h(C) = $\sqrt{(10-8)^2+(10-3)^2}$ = 7,28 ; el valor de f(C) = 10,2 + 7,28 = 17,48



El nodo E es el de menor valor por lo que se despliega su sucesor que es el nodo H; se calcula el valor de g(H) = 4,12 + 2,83 + $\sqrt{(6-2)^2+(6-4)^2}$ = 4,12 + 2,83 + 4,47 = 11,42 ; el valor h(H) = $\sqrt{(8-2)^2+(10-6)^2}$= 7,21 ; el valor de f(H) = 11,42 + 7,21 = 18,63.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4



El nodo D es el que tiene el menor valor por lo tanto se abren los sucesores que son F y G. El cálculo de g(F) = 7,84 + $\sqrt{(15-10)^2+(6-5)^2}$ = 7,84 + 5,10 = 12,94 y el valor de h(F) = $\sqrt{(10-8)^2+(10-6)^2}$ = 4,47 ; el cálculo de f(F) = 12,94 + 4,47 = 17,41. El valor de g(G) = 7,84 + $\sqrt{(15-8)^2+(10-5)^2}$ = 8,60; el valor de h(G) = 0 y por último el valor de f(G) = 8,60 + 0 = 8,60. En este caso como el menor valor es el del nodo G se ha llegado a la solución y el camino es A-C-D-G. Si hubiesen existido otros nodos que tenían menor valor y que se podía llegar a G se debía continuar con el cálculo hasta encontrar el de menor valor. En la búsqueda A* se puede retroceder del hijo hacia el padre cuando se buscan los sucesores, en el caso que se ha analizado esto no sucede porque existe dirección en los enlaces.

3.3 Optimización

La optimización es un proceso esencial en el ámbito de la inteligencia artificial (IA). Se refiere a la tarea de seleccionar la solución más adecuada de entre un conjunto de alternativas posibles. Esta elección se basa en criterios específicos, que pueden variar según el problema en cuestión. Por ejemplo, en un problema de planificación, la optimización podría buscar la ruta más corta o eficiente; mientras que, en un problema de aprendizaje automático, podría tratarse de minimizar el error en las predicciones.

La optimización no solo se limita a encontrar la "mejor" solución, sino que también puede involucrar la búsqueda de soluciones que sean "suficientemente buenas" en un tiempo razonable, especialmente cuando se trata de problemas complejos o de gran escala.

En el contexto de la IA, la optimización es fundamental, debido a que permite a los sistemas tomar decisiones informadas, mejorar su rendimiento y adaptarse a

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

situaciones cambiantes. A través de técnicas y algoritmos especializados, la IA puede explorar y evaluar diferentes soluciones, identificando aquellas que maximizan o minimizan un objetivo determinado.

3.3.1 Búsqueda local

La búsqueda local es una técnica de optimización que opera en el espacio de soluciones de un problema, manteniendo un único nodo o solución en cada iteración. A diferencia de otros algoritmos de búsqueda que exploran ampliamente el espacio de estados, la búsqueda local se centra en explorar el entorno inmediato de la solución actual, moviéndose iterativamente hacia soluciones vecinas que mejoren el criterio de optimización (Michiels et al., 2007).

Esta estrategia es especialmente útil para problemas con espacios de estados muy grandes o incluso infinitos, donde una búsqueda exhaustiva sería inviable. En lugar de buscar un camino específico, como en la resolución de laberintos, la búsqueda local se orienta a encontrar respuestas óptimas o subóptimas a preguntas complejas. Ejemplos de estas preguntas podrían ser: "¿Cuál es la configuración más eficiente en la conducción de un automóvil autónomo?" o "¿Cómo se optimiza la topología de una red de datos?".

Es importante destacar que, debido a su naturaleza, la búsqueda local no garantiza encontrar la solución óptima global. Sin embargo, tiene la ventaja de ser computacionalmente eficiente y, en muchos casos, proporciona soluciones "suficientemente buenas" en un tiempo razonable. Esta característica la hace valiosa en aplicaciones prácticas donde el tiempo y los recursos computacionales son limitados.

Se define los siguientes términos en la búsqueda local:

Función Objetivo: Es la función que determina la calidad o el valor de una solución en el espacio de búsqueda. El objetivo principal es maximizar esta función para obtener la solución más óptima, es decir, encontrar un máximo global.

Función de Costo: A diferencia de la función objetivo, la función de costo se centra en minimizar el valor asociado a una solución. El propósito es reducir al mínimo el costo asociado a una solución, buscando así un mínimo global.

Estado Actual: Representa la solución o nodo que el algoritmo está evaluando en un momento dado. Es el punto de referencia desde el cual se exploran las soluciones vecinas.

Estado Vecino: Son aquellos estados o soluciones que están inmediatamente

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

cercanos o relacionados con el estado actual. En términos prácticos, un estado vecino se obtiene realizando pequeñas modificaciones al estado actual. Estos estados ofrecen alternativas ligeramente diferentes y son esenciales para la exploración y optimización en la búsqueda local.

El mecanismo fundamental de los algoritmos de búsqueda local radica en iniciar con un estado actual y evaluar sus vecinos para determinar cuál de ellos ofrece una mejor solución, ya sea maximizando la función objetivo o minimizando la función de costo. El algoritmo se desplaza iterativamente hacia el mejor estado vecino y repite el proceso hasta que se cumpla un criterio de parada, como encontrar una solución que no pueda ser mejorada o alcanzar un número máximo de iteraciones.

Esta metodología permite explorar de manera eficiente el espacio de soluciones, especialmente en problemas complejos donde una búsqueda exhaustiva sería computacionalmente costosa.

Un ejemplo clásico de búsqueda local en inteligencia artificial es el problema del Viajante de Comercio (TSP, por sus siglas en inglés). En este problema, un viajante debe visitar una serie de ciudades exactamente una vez y regresar al punto de partida, minimizando la distancia total recorrida.

Supóngase que se tienen 4 ciudades (A, B, C, D) y las distancias entre ellas son las siguientes:

A-B: 10 km

A-C: 15 km

A-D: 20 km

B-C: 35 km

B-D: 20 km

C-D: 30 km

Pasos:

1) **Inicialización:** Se elige una ruta inicial aleatoria, por ejemplo, A -> B -> C -> D -> A.
2) **Función de Costo:** La suma de las distancias entre las ciudades en la ruta.

   Estado Actual: A -> B -> C -> D -> A con un costo de 10 (A-B) + 35 (B-C) + 30 (C-D) + 20 (D-A) = 95 km.
3) **Estados Vecinos:** Se genera rutas vecinas intercambiando dos ciudades en la ruta

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

actual. Por ejemplo, intercambiando B y C se obtiene A -> C -> B -> D -> A.

4) **Evaluación de Vecinos:** Calculamos el costo de la nueva ruta: 15 (A-C) + 35 (C-B) + 20 (B-D) + 20 (D-A) = 90 km.
5) **Selección del Mejor Vecino:** Comparamos el costo de la nueva ruta con el costo del estado actual. Si es menor, actualizamos el estado actual con el nuevo estado.
6) **Iteración:** Repetimos los pasos 4-6 hasta que no encontremos una mejora o hasta que alcancemos un número máximo de iteraciones.

Resultado:

Al final del proceso, se tendría una ruta que es una solución aproximada al problema del Viajante de Comercio. Es importante tener en cuenta que la búsqueda local podría no encontrar la solución óptima, pero generalmente encontrará una solución que es lo suficientemente buena en un tiempo razonable.

3.3.2 Búsqueda de escalada de colinas

La escalada de colinas (Hill Climbing) es una variante especializada de los algoritmos de búsqueda local, diseñada para encontrar soluciones óptimas en un espacio de estados (Minsky, 1961). En este enfoque, se parte de un estado inicial y se evalúan los estados vecinos utilizando una función específica, que puede ser una función objetivo para maximizar o una función de costo para minimizar. Si se encuentra un estado vecino que es mejor que el estado actual según la función de evaluación, se realiza un cambio, y ese estado vecino se convierte en el nuevo estado actual. Este proceso se repite iterativamente hasta que se alcanza un estado en el que ningún vecino ofrece una mejora, lo que se conoce como un máximo local en el caso de maximización o un mínimo local en el caso de minimización.

Un algoritmo de escalada de colinas se verá de la siguiente manera en pseudocódigo:

```
Función escaladaDeColinas(problema):
  actual = estadoInicialDelProblema()
  repetir:
    vecino = mejorVecino(actual)
    si valor(vecino) <= valor(actual):
      retornar actual
    actual = vecino
```

En este algoritmo, se inicia con un *estado actual*, en algunos problemas se verá

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

cuál es el estado actual, mientras que en otros se tiene que comenzar seleccionando uno aleatoriamente. Luego, se repite las siguientes acciones: se evalúa a los vecinos, seleccionando el de mejor valor. Luego, se compara el valor de este vecino con el valor del estado actual, si el vecino es mejor, se cambia el estado actual al estado vecino y luego se repite el proceso. El proceso finaliza cuando se compara el mejor vecino con el estado actual y el estado actual es mejor que el estado vecino. Luego, se devuelve el estado actual.

3.3.2.1 Mínimos y máximos locales y globales

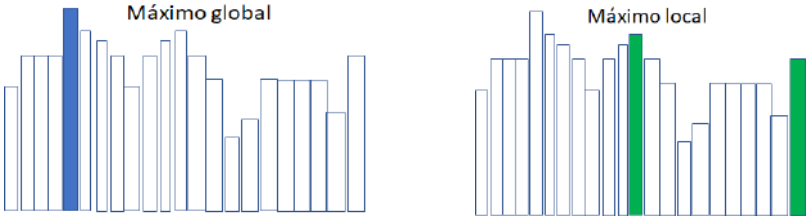


Figura 3.5 Máximo global y Máximo local

Elaborado por los Autores

Un algoritmo de escalada puede quedarse atascado en máximos o mínimos locales. Un máximo global (figura 3.5) es un estado que tiene el valor más alto de todos los estados en el espacio de estado. Un máximo local (figura 3.5) es un estado que tiene un valor más alto que sus estados vecinos, pero más bajo que el máximo global.

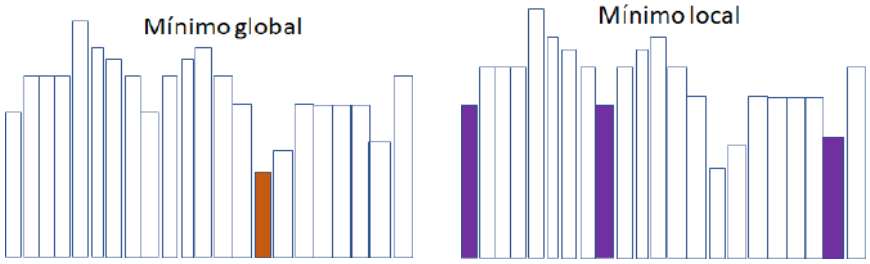


Figura 3.6 Mínimo global y Mínimo local

Elaborado por los Autores

Un mínimo global (figura 3.6) es un estado que tiene el valor más bajo de todos los estados en el espacio de estado. Un mínimo local (figura 3.6) es un estado que tiene un valor más bajo que sus estados vecinos.

El problema con los algoritmos de escalada de colinas es que pueden terminar en mínimos y máximos locales. Una vez que el algoritmo alcanza un punto cuyos vecinos son peores, para el propósito de la función, que el estado actual, el algoritmo se detiene.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Los tipos especiales de máximos y mínimos locales incluyen el **máximo local plano** (figura 3.7) **y el mínimo local plano**, donde varios estados de igual valor son adyacentes, formando una **meseta** cuyos vecinos tienen un valor peor debido a que la función de evaluación es plana constituyendo un máximo local plano, y el **hombro** (figura 3.7), donde varios estados de igual valor son adyacentes y los vecinos de la meseta puede ser tanto mejor como peor. A partir de la mitad de la meseta, el algoritmo no podrá avanzar en ninguna dirección.

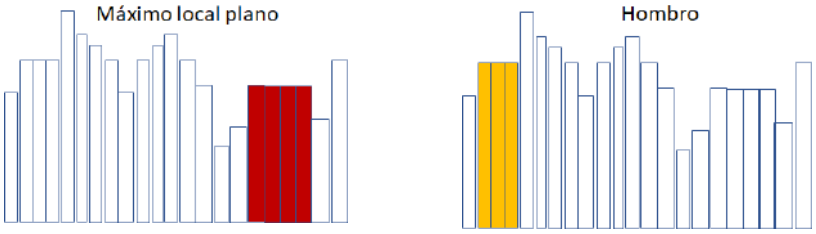


Figura 3.7 Máximo local plano y hombro

Elaborado por los Autores

Un ejemplo muy conocido de la utilización de la búsqueda de escalada de colinas en inteligencia artificial es el problema de las N-Reinas. El objetivo es colocar N reinas en un tablero de ajedrez de N x N de tal manera que ninguna reina pueda atacar a otra. En otras palabras, no debe haber dos reinas en la misma fila, columna o diagonal.

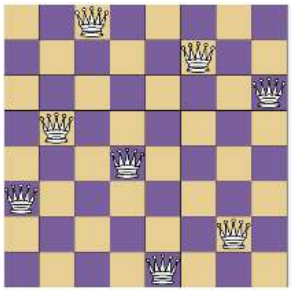


Figura 3.8 N-Reinas

Fuente: Mi diario Python. Luis Salcedo

El algoritmo de escalada de colinas para el Problema de las N-Reinas es el siguiente:

1. **Inicialización:** Coloca las N reinas en posiciones aleatorias en el tablero.
2. **Evaluación del Estado Actual:** Calcula el número de pares de reinas que se atacan mutuamente. Este será la función objetivo que se intentará minimizar.
3. **Bucle Principal:**

   Generar Vecinos: Para cada reina, mueve su posición en su columna actual para

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

generar nuevos tableros vecinos.

Evaluar Vecinos: Utiliza la función objetivo para evaluar qué tan bueno es cada tablero vecino.

Seleccionar el Mejor Vecino: Escoge el tablero vecino con el menor número de pares de reinas en conflicto.

Comparar con el Estado Actual:

- Si el mejor vecino es mejor que el estado actual, mueve a ese estado.
- Si no hay mejora, el algoritmo se detiene.

4. **Resultado:** El estado final es una solución al problema, aunque no necesariamente la mejor solución posible.

Este algoritmo es bastante simple y rápido, su problema es que puede quedarse detenido en óptimos locales, es decir, soluciones que son mejores que sus vecinos inmediatos, pero no son las mejores soluciones posibles.

La Búsqueda de Escalada de Colinas es utiliza amplaimente en problemas de optimización y búsqueda heurística, como el diseño de redes neuronales, la planificación de rutas y la asignación de tareas (Haykin, 2009).

3.3.3 Variantes de escalada de colinas

Debido a las limitaciones de la búsqueda de escalada de colinas, se han pensado múltiples variantes para superar el problema de quedarse atascado en mínimos y máximos locales. Lo que todas las variaciones del algoritmo tienen en común es que, sin importar la estrategia, cada uno todavía tiene el potencial de terminar en mínimos y máximos locales y no hay medios para continuar optimizando. Los algoritmos a continuación están redactados de tal manera que un valor más alto es mejor, pero también se aplican a las funciones de costo, donde el objetivo es minimizar el costo.

• **Ascenso más pronunciado:** Enfatiza la selección del vecino con la mayor mejora en el valor de aptitud (encontrar el máximo de una función) entre todos los vecinos generados en cada paso. Esta variante se centra en realizar el movimiento cuesta arriba más significativo en el paisaje de aptitud. Si bien puede ser efectiva para encontrar rápidamente un máximo local, también puede quedar atrapada en regiones estrechas y empinadas del paisaje.

Funcionamiento:

1) Comenzar con un estado inicial.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

2) Evaluar la función de evaluación (o función de costo) del estado actual.
3) Generar y evaluar todos los vecinos del estado actual.
4) Seleccionar el vecino que tiene el mayor incremento en la función de evaluación en comparación con el estado actual.
5) Si este vecino mejor tiene una evaluación más alta que el estado actual, moverse a ese estado y repetir desde el paso 2.
6) Si ningún vecino tiene una evaluación más alta que el estado actual, se ha encontrado un máximo local y el algoritmo termina.

Pseudocódigo:

```
Función AscensoMásPronunciado(estadoInicial):
  estadoActual = estadoInicial
  mientras no se alcance un máximo local o se alcance el número máximo
  de iteraciones:
    vecinos = generarVecinos(estadoActual)
    mejorVecino = seleccionarMejorVecino(vecinos)
    si el mejorVecino tiene un valor de función objetivo mayor que el
    estadoActual:
      estadoActual = mejorVecino
    sino:
        detenerse  #Alcanzamos un máximo local o una meseta
  devolver estadoActual
```

El pseudocódigo presentado ilustra el núcleo del proceso de Ascenso más pronunciado. Empieza con un estado inicial y, en cada ciclo, examina los estados circundantes para determinar si alguno supera la función objetivo. Si identifica un vecino más óptimo, el algoritmo se desplaza hacia ese estado y continua el proceso. Esta dinámica persiste hasta que se llega a un máximo local (donde no existe vecinos mejores) o se cumple un límite preestablecido de iteraciones.

• **Estocástico:** Se elije aleatoriamente entre los vecinos de mayor valor. Al hacer esto, se prefiere ir a cualquier dirección que mejore nuestro valor. Esto tiene sentido si, por ejemplo, el vecino de mayor valor conduce a un máximo local mientras que otro vecino conduce a un máximo global. Esto introduce un elemento de aleatoriedad que puede ayudar a escapar de óptimos locales. La escalada de colinas estocástica puede ser especialmente útil en búsquedas que tienen muchos máximos locales.

Funcionamiento:

1) Comenzar con un estado inicial.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

2) Evaluar la aptitud (o función de costo) del estado actual.
3) Generar una lista de todos los vecinos que tienen una mejor aptitud que el estado actual.
4) Seleccionar aleatoriamente uno de estos vecinos y moverse a ese estado.
5) Repetir los pasos 2-4 hasta que se cumpla un criterio de parada, como un número máximo de iteraciones sin mejora o si no se encuentran vecinos mejores.

Pseudocódigo:

```
Función Estocástico(estadoInicial, iteraciones):
  estadoActual = estadoInicial
  para i en rango(iteraciones):
    vecinoAleatorio = seleccionarVecinoAleatorio(estadoActual)
    si evaluar(vecinoAleatorio) > evaluar(estadoActual):
      estadoActual = vecinoAleatorio
  devolver estadoActual
```

Este pseudocódigo detalla un algoritmo estocástico básico. Inicia con un estado inicial y ejecuta un número preestablecido de iteraciones. Durante cada iteración, selecciona aleatoriamente un vecino del estado actual y evalúa si este vecino aleatorio es superior al estado actual de acuerdo con la función objetivo. En caso de ser más óptimo, el estado actual se reemplaza por el vecino aleatorio. Este procedimiento continúa durante un número determinado de iteraciones.

• **Primera elección:** elige el primer vecino de mayor valor, sin necesidad de evaluar a todos los vecinos, comparado con el estado actual que se generó aleatoriamente. Tan pronto como se encuentra un vecino que es mejor que el estado actual, se realiza el cambio. Los vecinos se suelen explorar en un orden aleatorio para evitar sesgos y para darle al algoritmo una mayor capacidad de exploración. Este algoritmo es útil cuando el número de vecinos es muy grande y evaluarlos todos sería computacionalmente costoso.

Funcionamiento:

1) Partir de un estado inicial.
2) Seleccionar aleatoriamente un vecino del estado actual.
3) Si este vecino mejora el estado actual según la función de evaluación, se adopta como nuevo estado actual.
4) Si no mejora, se selecciona otro vecino aleatoriamente y se repite el proceso.
5) El algoritmo termina cuando se ha explorado un número determinado de vecinos

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

sin encontrar mejora o cuando se cumple otro criterio de parada.

Pseudocódigo:

```
Función PrimeraElección(estadoInicial, iteraciones):
    estadoActual = estadoInicial
    para i en rango(iteraciones):
        vecinoAleatorio = seleccionarVecinoAleatorio(estadoActual)
        si evaluar(vecinoAleatorio) > evaluar(estadoActual):
            estadoActual = vecinoAleatorio
            si evaluación(estadoActual) >= valorMeta:
                devolver estadoActual  # Se alcanzó una solución aceptable
    devolver estadoActual  # Se alcanzó el número máximo de iteraciones
```

Este pseudocódigo ilustra el funcionamiento del algoritmo de Primera Elección, que es una variante de la búsqueda local. Comienza desde un estado inicial y realiza un número especificado de iteraciones. En cada iteración, selecciona un vecino aleatorio del estado actual y evalúa si el vecino aleatorio es mejor que el estado actual según la función objetivo. En caso de ser superior, el estado actual se actualiza al vecino aleatorio. El proceso se repite hasta que se alcanza una solución aceptable (según un criterio de parada) o hasta que se alcanza el número máximo de iteraciones permitidas.

• **Algoritmos Genéticos:** Estos algoritmos están inspirados en el proceso natural de la evolución biológica, utilizan una población de soluciones y aplican operaciones inspiradas en la genética, como la mutación y el cruce, para explorar el espacio de estados (Pham & Karaboga, 2012). Tienen la capacidad de explorar un amplio espacio de soluciones y encontrar soluciones óptimas o subóptimas en problemas complejos.

Funcionamiento:

1) Inicializar una población de individuos de manera aleatoria.
2) Evaluar la aptitud de cada individuo en la población.
3) Seleccionar individuos para la reproducción basados en su aptitud.
4) Aplicar operadores genéticos como el cruzamiento y la mutación para generar una nueva población.
5) Repetir los pasos 2-4 hasta que se cumpla un criterio de parada, como un número máximo de generaciones o una aptitud objetivo.

Pseudocódigo:

```
AlgoritmoGenético(tamañoPoblación, tasaMutación, generaciones):
    población = inicializarPoblaciónAleatoria(tamañoPoblación)
    para generación en rango(generaciones):
```

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

```
    evaluación = evaluarPoblación(población)
    nuevaPoblación = []
  mientras tamaño(nuevaPoblación) < tamañoPoblación:
      padre1 = seleccionarPadre(población, evaluación)
      padre2 = seleccionarPadre(población, evaluación)
      hijo = cruzar(padre1, padre2)
      si aleatorio() < tasaMutación:
        mutar(hijo)
      agregar hijo a nuevaPoblación
    población = nuevaPoblación
  devolver el mejor individuo de la población
```

El pseudocódigo describe un algoritmo genético. A continuación, se describen las funciones principales del mismo:

- *inicializarPoblaciónAleatoria(tamañoPoblación)*: Crea una población inicial de individuos de manera aleatoria.
- *evaluarPoblación(población):* Evalúa cada individuo en la población utilizando una función de aptitud y devuelve sus valores de aptitud.
- *seleccionarPadre(población, evaluación):* Selecciona un individuo para ser padre basado en su aptitud relativa.
- *cruzar(padre1, padre2):* Realiza la cruza (reproducción) entre dos padres para generar un hijo.
- *mutar(hijo):* Aplica una mutación al hijo, posiblemente cambiando uno o más genes.
- *aleatorio():* Genera un valor aleatorio entre 0 y 1.

El algoritmo genético se ejecuta durante un número especificado de generaciones. En cada generación, los individuos se evalúan en función de su aptitud, se seleccionan padres para reproducción, se realiza la cruza y, en algunos casos, se aplica una mutación. El proceso se repite hasta alcanzar un número fijo de generaciones.

• **Reinicio aleatorio:** Esta técnica ejecuta la escalada de colinas múltiples veces desde distintos estados iniciales aleatorios. Su objetivo principal es superar la limitación de la escalada de colinas tradicional que tiende a estancarse en máximos (o mínimos) locales. En vez de confinarse a una sola búsqueda desde un punto inicial, se exploran diversas soluciones partiendo de diferentes estados. Tras repetir este proceso en numerosas ocasiones, se opta por la solución más óptima encontrada entre todas las iteraciones.

Funcionamiento:

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

1) Establecer un número máximo de reinicios.
2) Para cada reinicio:
   a. Seleccionar un estado inicial al azar.
   b. Realizar la escalada de colinas desde ese estado inicial hasta que se encuentre un máximo local.
   c. Guardar la solución si es mejor que las soluciones anteriores.
3) Al final de todos los reinicios, devolver la mejor solución encontrada.

Pseudocódigo:

```
ReinicioAleatorio(iteraciones, tasaReinicio):
  mejorSolucion = SolucionAleatoria()  # Inicializar con una solución aleatoria
  para i en rango(iteraciones):
    solucionActual = MejorVecino(mejorSolucion)  # Encuentra el mejor vecino
    si solucionActual es mejor que mejorSolucion:
      mejorSolucion = solucionActual  # Actualiza la mejor solución
    si aleatorio() < tasaReinicio:
      mejorSolucion = SolucionAleatoria()  # Reinicia aleatoriamente
  devolver mejorSolucion
```

A continuación, se describen las funciones y componentes clave del pseudocódigo de un algoritmo de Reinicio Aleatorio.

iteraciones: Es el número máximo de iteraciones que se ejecutarán.

tasaReinicio: La probabilidad de realizar un reinicio aleatorio en cada iteración.

SolucionAleatoria(): Se genera una solución inicial aleatoria.

MejorVecino(solucion): Encuentra el mejor vecino de la solución actual. Esto implica explorar soluciones cercanas a la actual y seleccionar la mejor.

aleatorio(): Genera un valor aleatorio entre 0 y 1.

El algoritmo empieza con una solución inicial aleatoria. Luego, realiza un bucle durante un número determinado de iteraciones. En cada iteración, busca el mejor vecino de la solución actual y compara si es mejor que la mejor solución conocida hasta ese instante. De ser el caso de que sea superior, actualiza la mejor solución. Además, en cada iteración, hay una probabilidad (tasaReinicio) de realizar un reinicio aleatorio, esto implica que la búsqueda se reinicia desde una solución aleatoria en lugar de continuar desde la solución actual.

• **Búsqueda de haz local:** A diferencia de los algoritmos de búsqueda local convencionales que trabajan con un único nodo, este enfoque mantiene y evalúa múltiples

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

estados simultáneamente. En cada iteración, selecciona los k vecinos más prometedores. Al operar con varias soluciones en paralelo, este método incrementa las probabilidades de hallar un óptimo global, evitando quedar atrapado en óptimos locales, un problema común en la escalada de colinas. No obstante, este beneficio viene acompañado de un mayor costo computacional, ya que se requiere evaluar varias soluciones en cada iteración (Russell & Norvig, 2021).

Funcionamiento:

1) Comienza con k estados iniciales generados aleatoriamente.
2) En cada iteración, se exploran todos los vecinos de todos los k estados en el haz.
3) De todos estos vecinos, se seleccionan los k mejores para formar el nuevo haz para la siguiente iteración.

El proceso se repite hasta que no se encuentren mejoras en el haz o hasta que se cumpla algún otro criterio de parada.

Pseudocódigo:

```
BúsquedaHazLocal(k, iteraciones):
  soluciones = GenerarKSolucionesIniciales(k)  # Genera k soluciones iniciales aleatorias
  para i en rango(iteraciones):
    nuevasSoluciones = Vacío()  # Inicializa una lista vacía para almacenar las nuevas soluciones
    para solucion en soluciones:
      vecinos = GenerarVecinos(solucion)  # Genera vecinos de la solución actual
      mejorVecino = MejorVecino(vecinos)  # Encuentra el mejor vecino
      agregar mejorVecino a nuevasSoluciones  # Agrega el mejor vecino a la lista de nuevas soluciones
      soluciones = SeleccionarKSolucionesTop(nuevasSoluciones, k)  # Selecciona las mejores k soluciones
    mejorSolucion = MejorSolucion(soluciones)  # Encuentra la mejor solución de todas
    devolver mejorSolucion
```

A continuación, se encuentra las funciones y componentes del pseudocódigo del algoritmo de Búsqueda de Haz Local.

k: El número de soluciones en el conjunto de haces.

iteraciones: El número máximo de iteraciones que se ejecutarán.

GenerarKSolucionesIniciales(k): Genera k soluciones iniciales aleatorias.

GenerarVecinos(solucion): Genera los vecinos de una solución dada.

MejorVecino(vecinos): Encuentra el mejor vecino entre un conjunto de vecinos.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

SeleccionarKSolucionesTop(soluciones, k): Selecciona las mejores k soluciones entre todas las soluciones generadas.

MejorSolucion(soluciones): Encuentra la mejor solución entre todas las soluciones finales.

Empieza el algoritmo generando k soluciones iniciales aleatorias. A continuación, se ejecuta un bucle durante un número determinado de iteraciones. En cada iteración, se genera vecinos para cada solución en el conjunto de haces, encuentra el mejor vecino y procede a agregarle a la lista de nuevas soluciones.

Al concluir cada iteración, se escogen las mejores k soluciones de la lista de nuevas soluciones y se actualiza el conjunto de haces con esas soluciones. Finalmente, después de todas las iteraciones, el algoritmo encuentra la mejor solución entre todas las soluciones finales y la devuelve como resultado.

• **Optimización por Enjambre de Partículas:** Esta técnica, inspirada en el comportamiento colectivo de aves y peces, utiliza "partículas" que representan soluciones en un espacio de búsqueda. A diferencia de la búsqueda de haz local, en la optimización por ejambre de particulas, las partículas ajustan su posición no solo en función de su propia experiencia, sino también de la de sus vecinos. Cada partícula posee una posición, que simboliza una solución potencial; una velocidad, que indica su movimiento; y una memoria, que almacena su mejor posición hallada. Una particularidad distintiva es la colaboración entre partículas, compartiendo información y orientando al enjambre hacia zonas del espacio de búsqueda que muestran ser prometedoras, todo ello fundamentado en las vivencias acumuladas por el conjunto de partículas (Benítez et al., 2014).

Funcionamiento:

1) Inicialización: Se crea un enjambre de partículas con posiciones y velocidades aleatorias.
2) Evaluación: Cada partícula evalúa su posición actual utilizando una función objetivo.
3) Actualización de la memoria: Si la posición actual de una partícula es mejor (según la función objetivo) que su mejor posición conocida, se actualiza su memoria.
4) Actualización de la velocidad y posición: La velocidad de cada partícula se ajusta en función de su memoria y de la mejor posición encontrada por todo el enjambre.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Luego, se actualiza la posición de la partícula basándose en su velocidad.

5) Iteración: Los pasos 2 a 4 se repiten hasta que se cumpla un criterio de parada, como un número máximo de iteraciones o una mejora mínima en la función objetivo.

Pseudocódigo:

```
OptimizaciónEjambrePartículas(iteraciones, tamañoEnjambre, c1, c2):
  enjambre = InicializarEnjambreAleatoriamente(tamañoEnjambre)  # Inicializa el enjambre con partículas aleatorias
  para i en rango(iteraciones):
    para cada partícula en enjambre:
      calcular la aptitud de la partícula  # Evalúa la calidad de la solución
      si la aptitud es mejor que la mejor aptitud personal de la partícula:
        actualizar la mejor aptitud personal de la partícula
        actualizar la mejor posición personal de la partícula
      encontrar la partícula con la mejor aptitud personal en todo el enjambre
      para cada partícula en enjambre:
        actualizar la velocidad y posición de la partícula:
          velocidad = velocidad + c1 * rand() * (mejor posición personal - posición actual)
          velocidad = velocidad + c2 * rand() * (mejor posición global - posición actual)
          posición = posición + velocidad
  encontrar la partícula con la mejor aptitud global en todo el enjambre
  devolver la mejor posición global encontrada
```

El pseudocódigo describe el algoritmo Optimización por Enjambre de Partículas, a continuación, se indican las variables y componentes importantes involucrados:

iteraciones: El número máximo de iteraciones que se ejecutarán.

tamañoEnjambre: El número de partículas en el enjambre.

c1 y c2: Coeficientes de aceleración que controlan la influencia de la mejor posición personal y la mejor posición global en el movimiento de las partículas.

InicializarEnjambreAleatoriamente(tamañoEnjambre): Inicializa el enjambre con posiciones y velocidades aleatorias.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

calcular la aptitud de la partícula: Evalúa la calidad de la solución representada por la partícula.

mejor posición personal: La mejor posición alcanzada por una partícula individual.

mejor posición global: La mejor posición alcanzada por cualquier partícula en el enjambre.

El algoritmo Optimización por Enjambre de Partículas inicia con un enjambre de partículas con posiciones y velocidades aleatorias. Durante un número determinado de iteraciones, cada partícula evalúa su aptitud y, si es superior a su mejor registro personal, actualiza su posición. Se identifica la partícula con la aptitud más destacada del enjambre. Posteriormente, se ajustan la velocidad y posición de cada partícula basándose en las mejores posiciones personales y globales, influenciadas por los coeficientes c1 y c2. Al concluir las iteraciones, el algoritmo determina la partícula con la mejor aptitud global, presentando su posición como la solución óptima o aproximada al problema de optimización.

• **Búsqueda tabú:** Esta técnica metaheurística de optimización se caracteriza por mantener una lista de los estados recientemente explorados, denominada "lista tabú", para prevenir su reexploración inmediata. Su objetivo es optimizar soluciones evitando quedar atrapado en óptimos locales. Para ello, no solo se apoya en su memoria a corto plazo, sino que también implementa estrategias de diversificación, que permiten abordar distintas regiones del espacio de soluciones, y estrategias de intensificación, que se centran en investigar detalladamente áreas con potencial prometedor (Benítez et al., 2014).

Funcionamiento:

1) Iniciar con una solución inicial y se evalúa su calidad.
2) En cada iteración, se generan soluciones vecinas y se selecciona la mejor, incluso si es peor que la solución actual.
3) La solución seleccionada se añade a la lista tabú.

Pseudocódigo:

```
BúsquedaTabú(iteraciones, tamañoListaTabú):
  soluciónActual = GenerarSoluciónInicial()  # Genera una solución inicial aleatoria
  mejorSolución = soluciónActual
  listaTabú = ListaVacia()
 para i en rango(iteraciones):
    vecindario = GenerarVecindario(soluciónActual)  # Genera un vecindario de soluciones vecinas
```

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

```
    mejorVecino = NULL
    for cada vecino en vecindario:
        si vecino no está en listaTabú y aptitud(vecino) < aptitud(mejorVecino):
            mejorVecino = vecino
    si mejorVecino es NULL:
        detener  # No se encontró un mejor vecino válido
    soluciónActual = mejorVecino
    si aptitud(mejorVecino) < aptitud(mejorSolución):
        mejorSolución = mejorVecino
    agregar mejorVecino a listaTabú
    si tamaño de listaTabú > tamañoListaTabú:
        eliminar el elemento más antiguo de listaTabú
devolver mejorSolución
```

A continuación, se describe el pseudocódigo del algoritmo de Búsqueda Tabú, sus variables y componentes importantes:

iteraciones: El número máximo de iteraciones que se ejecutarán.

tamañoListaTabú: El tamaño máximo de la lista tabú que almacena soluciones prohibidas recientes.

GenerarSoluciónInicial(): Genera una solución inicial aleatoria.

GenerarVecindario(soluciónActual): Genera un vecindario de soluciones vecinas a partir de la solución actual.

aptitud(solución): Evalúa la calidad de una solución dada (menos es mejor).

ListaVacia(): Inicializa una lista tabú vacía.

El algoritmo inicia con una solución aleatoria, considerándola como la mejor hasta ese momento. En cada iteración, genera soluciones vecinas y selecciona el mejor vecino que no este en la lista tabú con aptitud superior. Si este vecino supera la mejor solución actual, se actualiza. Este vecino se añade a la lista tabú, eliminando el elemento más antiguo si se excede el tamaño máximo de la lista. Tras un número determinado de iteraciones, el algoritmo presenta la mejor solución hallada como la respuesta óptima o aproximada al problema de optimización.

3.3.4 Recocido simulado

El algoritmo de Recocido Simulado, también conocido como Simulated Annealing en inglés, se destaca como una estrategia robusta para abordar el problema de quedar atrapado en óptimos locales en la búsqueda de escalada de colinas. Su concepto se inspira en la metalurgia, donde se calienta un metal y luego se permite que se enfríe

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

gradualmente para mejorar sus propiedades mecánicas. De manera análoga, el algoritmo de Recocido Simulado opera en la búsqueda de soluciones óptimas.

El proceso comienza con el algoritmo funcionando a una "temperatura" alta, lo que implica que tiene una alta probabilidad de tomar decisiones aleatorias. A medida que esta "temperatura" disminuye durante el proceso, el algoritmo se vuelve más "selectivo", es decir, reduce la probabilidad de tomar decisiones aleatorias. Este mecanismo de enfriamiento controlado es fundamental, ya que permite al algoritmo explorar estados vecinos que podrían ser peores que el estado actual, lo que evita quedar atrapado en máximos locales.

A continuación, se presenta un pseudocódigo para el algoritmo de Recocido Simulado:

```
función RecocidoSimulado(problema, máx):
  actual = estadoInicial del problema
  para t de 1 a máx:
    T = Temperatura(t)
    vecino = vecinoAleatorio del actual
    ΔE = diferencia de aptitud entre el vecino y el estado actual
    si ΔE > 0:
      actual = vecino
    sino, con probabilidad e^(ΔE/T) conjunto actual = vecino
  retornar actual
```

Este algoritmo toma como entrada un problema y el número máximo de iteraciones a realizar. En cada iteración, se establece la temperatura T mediante una función de temperatura. Esta función devuelve un valor más alto en las primeras iteraciones (cuando t es bajo) y un valor más bajo en las iteraciones posteriores (cuando t es alto). Luego, se selecciona un vecino aleatorio y se calcula ΔE, que mide cuánto mejor es el vecino en comparación con el estado actual.

Si ΔE es positivo, lo que significa que el vecino es mejor que el estado actual, se establece el estado actual como el vecino. Sin embargo, cuando ΔE es negativo, lo que indica que el vecino es peor, aún existe la posibilidad de que el estado actual se establezca como el vecino. Esta probabilidad se calcula mediante la fórmula $e^{(\Delta E/T)}$. Cuanto más negativo sea ΔE, menor será la probabilidad de seleccionar el vecino, y cuanto mayor sea la temperatura T, mayor será la probabilidad de aceptar un vecino peor.

El Recocido Simulado es una estrategia eficaz para explorar soluciones en problemas de optimización, permitiendo movimientos hacia estados peores en busca de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

una solución global óptima (Russell & Norvig, 2021). A medida que disminuye la temperatura, se vuelve más selectivo y tiende a converger hacia la mejor solución encontrada hasta el momento. Esta propiedad lo hace valioso en la resolución de problemas complejos donde se desea evitar quedar atrapado en óptimos locales.

Un ejemplo interesante de la aplicación del algoritmo de Recocido Simulado en Inteligencia Artificial es el problema de la Mochila. En este problema, se tiene una mochila con una capacidad de peso máxima y un conjunto de objetos, cada uno con un peso y un valor. El objetivo es seleccionar un subconjunto de objetos de tal manera que el peso total no exceda la capacidad de la mochila y el valor total sea máximo.

Pseudocódigo de Recocido Simulado para el Problema de la Mochila.

1. Inicializar solución_actual con una selección aleatoria de objetos
2. Inicializar mejor_solución con solución_actual
3. Inicializar temperatura = 1000 (valor alto)
4. Inicializar factor_enfriamiento = 0.99

Mientras temperatura > 1:

1. Generar solución_vecina cambiando un objeto aleatorio en solucion_actual
2. Calcular delta_valor = valor(solución_vecina) - valor(solución_actual)
3. Si delta_valor > 0:
   - Actualizar solución_actual = solución_vecina
   - Si valor(solución_actual) > valor(mejor_solución):
     - Actualizar mejor_solución = solución_actual
4. Si no:
   - Generar un número aleatorio r entre 0 y 1
   - Si r < exp(delta_valor / temperatura):
     - Actualizar solución_actual = solución_vecina
5. Reducir la temperatura: temperatura *= factor_enfriamiento

Explicación:

Inicialización: Se comienza con una solución inicial aleatoria y una temperatura alta.

Bucle Principal: El algoritmo entra en un bucle donde se generan soluciones vecinas y se evalúan.

Generación de Vecinos: Se crea una solución vecina cambiando un objeto aleatorio en la solución actual.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Evaluación de Valor: Se calcula la diferencia de "valor" entre la solución vecina y la solución actual.

Actualización de Solución: Si la solución vecina es mejor (mayor valor), se acepta como la nueva solución actual. Si es peor, podría aceptarla con una probabilidad que disminuye con la temperatura.

Enfriamiento: Se reduce la temperatura según un factor de enfriamiento y repite el proceso.

Al final del algoritmo, mejor_solución contendrá la mejor solución encontrada para el problema de la mochila.

Este algoritmo es especialmente útil en problemas de optimización combinatoria y ha demostrado su eficacia en una variedad de aplicaciones prácticas en inteligencia artificial.

El algoritmo de Recocido Simulado tiene diversas aplicaciones en el campo de la Inteligencia Artificial y la optimización. A continuación, estas son algunas de las áreas donde este algoritmo es especialmente útil:

1. Optimización Combinatoria

   Problema del Viajante de Comercio (TSP)

   Problema de la Mochila

   Problema de Asignación de Tareas

2. Aprendizaje Automático

   Selección de Características: Para mejorar la eficiencia de los modelos de aprendizaje automático.

   Optimización de Hiperparámetros: Para encontrar la mejor configuración de un modelo de aprendizaje automático.

3. Robótica

   Planificación de Rutas para Robots: Optimización de la trayectoria que debe seguir un robot para completar una tarea.

4. Redes de Computadoras

   Balanceo de Carga: Para distribuir eficientemente las tareas entre diferentes servidores.

   Optimización de Redes: Para encontrar la ruta más eficiente para el envío de paquetes de datos.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

5. Logística y Cadena de Suministro

Optimización de Rutas de Entrega: Para encontrar la ruta más eficiente para la entrega de productos.

6. Diseño VLSI (Very Large Scale Integration)

Ubicación de Componentes: Para optimizar la disposición de componentes en un chip.

7. Bioinformática

Alineación de Secuencias: Para encontrar la mejor alineación entre dos secuencias de ADN, ARN o proteínas.

8. Juegos y Estrategias

Resolución de Puzzles y Juegos de Tablero: Como el Sudoku o el problema de las N-reinas.

9. Energía y Medio Ambiente

Optimización de Sistemas de Energía: Para encontrar la configuración más eficiente en sistemas de energía renovable.

10. Finanzas

Optimización de Portafolios: Para encontrar la combinación óptima de activos financieros.

El Recocido Simulado es un algoritmo versátil que se adapta bien a problemas que tienen un gran espacio de soluciones y donde se busca una solución óptima o cercana a la óptima en un tiempo razonable.

3.3.5 Programación lineal

La programación lineal es una familia de problemas que optimizan una ecuación lineal (una ecuación de la forma $y = ax_1 + bx_2 + \ldots$).

La programación lineal tendrá los siguientes componentes:

• Una función de costo que se quiere minimizar: $c_1x_1 + c_2x_2 + \ldots + c_nx_n$. Aquí, cada x_- es una variable y está asociada con algún costo c_-.

• Una restricción que se representa como una suma de variables que es menor o igual a un valor ($a_1x_1 + a_2x_2 + \ldots + a_nx_n \leq b$) o exactamente igual a este valor ($a_1x_1 + a_2x_2 + \ldots + a_nx_n = b$). En este caso, x_- es una variable, y a_- es algún recurso asociado a ella, y b es cuántos recursos se puede dedicar a este problema.

• Límites individuales en las variables (por ejemplo, que una variable no puede ser negativa) de la forma $l_i \leq x_i \leq u_i$.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Considere el siguiente ejemplo:

• Dos máquinas, X_1 y X_2. X_1 cuesta \$40/hora para funcionar, X_2 cuesta \$70/hora para funcionar. El objetivo es minimizar el costo. Esto se puede formalizar como una función de costo: $40x_1 + 70x_2$.

• X_1 requiere 6 unidades de trabajo por hora. X_2 requiere 3 unidades de trabajo por hora. Total de 22 unidades de mano de obra a gastar. Esto se puede formalizar como una restricción: $6x_1 + 3x_2 \leq 22$.

• X_1 produce 10 unidades de salida por hora. X_2 produce 14 unidades de salida por hora. La empresa necesita 95 unidades de producción. Esta es otra restricción. Literalmente, se puede reescribir como $10x_1 + 14x_2 \geq 95$. Sin embargo, las restricciones deben tener la forma ($a_1x_1 + a_2x_2 + \ldots + a_nx_n \leq b$) o ($a_1x_1 + a_2x_2 + \ldots + a_nx_n = b$). Por lo tanto, multiplicamos por (-1) para obtener una ecuación equivalente de la forma deseada: $(-10x_1) + (-14x_2) \leq -95$.

Un algoritmo de optimización para programación lineal requiere conocimientos previos en geometría y álgebra lineal que no queremos asumir, en su lugar, se puede usar algoritmos que ya existen en diferentes lenguajes.

3.3.6 Satisfacción de restricciones

Los problemas de satisfacción de restricciones tienen un conjunto de variables, un conjunto de dominios (cada uno asociado a una variable) y un conjunto de restricciones. Son una clase de problemas en los que es necesario asignar valores a las variables mientras se satisfacen algunas series de restricciones o condiciones.

ESTUDIANTES	CURSOS QUE TOMA	DIAS DIPOSIBLES PARA EXAMENES
1	A B C	Lunes Martes Miércoles
2	B D E	Lunes Martes Miércoles
3	C E F	Lunes Martes Miércoles
4	E F G	Lunes Martes Miércoles

Figura 3.9 Cursos que toman los estudiantes

Elaborado por los Autores

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Los problemas de satisfacción de restricciones tienen las siguientes propiedades:

• Conjunto de variables (x_1, x_2, …, x_n)

• Conjunto de dominios para cada variable {D_1, D_2, …, D_n}

• Conjunto de restricciones C

Considere el siguiente ejemplo (figura 3.9) en que cada uno de los estudiantes 1,2,3,4 está tomando tres cursos de los siguientes A, B, C, D, E, F, G. Cada curso debe tener un examen, y los días posibles para los exámenes son los lunes, martes y miércoles. Sin embargo, el mismo estudiante no puede tener dos exámenes el mismo día. En este caso, las variables son los cursos, el dominio son los días y las restricciones son qué cursos no se pueden programar para tener un examen el mismo día porque los está tomando el mismo estudiante.

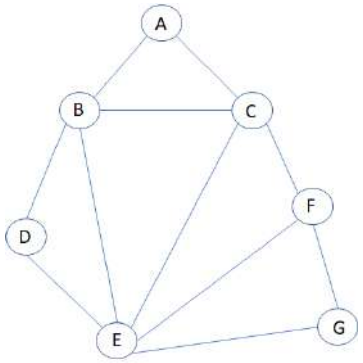


Figura 3.10 Grafo de los cursos

Elaborado por los Autores

Este problema se puede resolver empleando restricciones que se representan como un gráfico. Cada nodo del gráfico (figura 3.10) es un curso distinto, y se pone la restricción entre dos cursos que es de que no se pueden programar para el mismo día.

Algunos términos que vale la pena conocer sobre los problemas de satisfacción de restricciones son:

• Una **Restricción Fuerte** es una restricción que debe ser satisfecha en una solución correcta.

• Una **Restricción Suave** es una restricción que expresa qué solución se prefiere sobre otras.

• Una **Restricción Unaria** es una restricción que involucra solo una variable. En nuestro ejemplo, una restricción unaria sería decir que el curso A no puede tener un examen el lunes {A ≠ lunes}.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

• Una **Restricción Binaria** es una restricción que involucra dos variables. Este es el tipo de restricción que usamos en el ejemplo, se dice que dos cursos no pueden tener el mismo valor {A ≠ B} con respecto al día del examen.

3.3.6.1 Consistencia de nodo

En el ámbito de la programación de restricciones, la "consistencia del nodo" se refiere a la condición en la que todos los valores en el dominio de una variable particular cumplen con las restricciones unarias asociadas a esa variable (Russell & Norvig, 2021). En otras palabras, cada valor que la variable puede tomar es viable según las restricciones impuestas sobre ella.

Para que un nodo (variable) sea consistente:

1. Todos los valores dentro de su dominio deben cumplir cualquier restricción unaria que se aplique a esa variable.
2. Si un valor en el dominio de una variable no cumple con la restricción unaria, ese valor se excluye del dominio.

Continuando con el ejemplo anterior, se tiene dos cursos, A y B. El dominio de cada curso es {lunes, martes, miércoles} y las restricciones son {A ≠ lunes, B ≠ martes, B ≠ lunes, A ≠ B}. Ahora, ni A ni B son consistentes, porque las restricciones existentes les impiden tomar todos los valores que están en su dominio. Sin embargo, si eliminamos el lunes del dominio de A, tendrá consistencia de nodo. Para lograr la consistencia del nodo en B, se tendrá que eliminar tanto el lunes como el martes de su dominio.

La verificación de la consistencia del nodo es un paso importante en algoritmos de programación de restricciones como el backtracking, debido a que ayuda a reducir el espacio de búsqueda y a eliminar asignaciones que no conducirán a una solución válida, optimizando así el proceso de resolución del problema (Shi, 2019).

3.3.6.2 Consistencia del arco

La consistencia de arco es cuando todos los valores en el dominio de una variable satisfacen las restricciones binarias de la variable. Por lo que se debe verificar y asegurar de que, para cada valor en el dominio de una variable, exista algún valor en el dominio de otra variable que satisface la restricción binaria entre ambas variables. En otras palabras, para hacer que X sea coherente con el arco con respecto a Y, elimine elementos del dominio de X hasta que cada opción para X tenga una opción posible para Y.

Para que un arco (una restricción binaria entre dos variables) sea consistente:

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

1. Para cada valor en el dominio de la primera variable, debe existir al menos un valor en el dominio de la segunda variable que satisface la restricción binaria entre ambas.
2. Si no existe tal valor para algún valor de la primera variable, ese valor se elimina del dominio de la primera variable.

Considérese el ejemplo de los estudiantes con los dominios revisados: A :{martes, miércoles} y B:{miércoles}. Para que A tenga coherencia del arco con B, sin importar el día en que se programe el examen de A (desde su dominio), B aún podrá programar un examen. ¿Es A arco-consistente con B? Si A toma el valor martes, entonces B puede tomar el valor miércoles. Sin embargo, si A toma el valor miércoles, entonces no hay valor que B pueda tomar (recuerde que una de las restricciones es A ≠ B). Por lo tanto, A no es arco-consistente con B. Para cambiar esto, se puede eliminar miércoles del dominio de A. Entonces, cualquier valor que tome A (el martes es la única opción) deja un valor para que lo tome B (miércoles). Ahora, A es consistente del arco con B. Veamos un algoritmo en pseudocódigo que hace que una variable sea consistente con el arco con respecto a alguna otra variable. En estos problemas, generalmente se tienen variables que pueden tomar ciertos valores y restricciones que limitan las combinaciones posibles de estos valores. El objetivo es encontrar una asignación de valores a las variables de manera que se satisfagan todas las restricciones.

```
función Revisar(restrición, X, Y):
  revisado = falso
  # Iterar sobre cada valor en el dominio de X
  para cada x en X.dominio:
    # Verificar si existe algún valor y en el dominio de Y que satisface la restricción
    si no existe y en Y.dominio tal que restricción(x, y) sea verdadero:
      eliminar x de X.dominio
      revisado = verdadero
  # Retornar si se hizo alguna modificación en el dominio de X
  retorne revisado
```

Este algoritmo comienza rastreando si se realizó algún cambio en el dominio de X, utilizando la variable revisado. Esto será útil en el próximo algoritmo que se examine. Luego, el código se repite para cada valor en el dominio de X y ve si Y tiene un valor que satisface las restricciones. Si es así, entonces no haga nada, si no, elimine este valor del dominio de X.

A menudo se está interesado en hacer que todo el problema sea consistente y no solo una variable con respecto a otra. En este caso, se utiliza un algoritmo llamado PM,

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

que usa Revisar:

```
función PM(restricción):
    cola = todos los arcos con la restricción
    mientras cola no esté vacía:
        (X, Y) = Quitar de la cola (cola)
        si Revisar(restricción, X, Y):
            si tamaño del dominio de X == 0:
                retorne falso
            para cada Z en X.vecinos - {Y}:
                Poner en cola (Z, X)
    retorne verdadero
```

Este algoritmo agrega todos los arcos del problema a una cola. Cada vez que considera un arco, lo elimina de la cola. Luego, ejecuta en el algoritmo Revisar para ver si este arco es consistente. Si se hicieron cambios para hacerlo consistente, se necesitan más acciones. Si el dominio resultante de X está vacío, significa que este problema de satisfacción de restricciones no tiene solución (debido a que no hay valores que X pueda tomar que permitan que Y tome cualquier valor dadas las restricciones). Si el problema no se considera irresoluble en el paso anterior, entonces, dado que se cambió el dominio de X, debemos ver si todos los arcos asociados con X aún son consistentes. Es decir, tomamos todos los vecinos de X excepto Y, y agregamos los arcos entre ellos y X a la cola. Sin embargo, si en el algoritmo Revisar devuelve falso, lo que significa que el dominio no cambió, simplemente continuamos considerando los otros arcos.

Si bien el algoritmo para la consistencia del arco puede simplificar el problema, no necesariamente lo resolverá, debido a que solo considera las restricciones binarias y no cómo se pueden interconectar varios nodos. Nuestro ejemplo anterior, donde cada uno de los 4 estudiantes está tomando 3 cursos, permanece sin cambios al ejecutar PM en él.

Un *problema de satisfacción de restricciones* puede verse como un problema de búsqueda:

• **Estado inicial:** asignación vacía (todas las variables no tienen ningún valor asignado).

• **Acciones:** agregue una {variable = valor} a la asignación; es decir, dar un valor a alguna variable.

• **Modelo de transición:** muestra cómo al agregar la tarea cambia la tarea. No hay mucha profundidad en esto: el modelo de transición devuelve el estado que incluye la asignación después de la última acción.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

• **Prueba de objetivo:** comprobar si a todas las variables se les asigna un valor y se cumplen todas las restricciones.

• **Función de coste del trayecto:** todos los trayectos tienen el mismo coste. Como se mencionó anteriormente, a diferencia de los típicos problemas de búsqueda, los problemas de optimización se preocupan por la solución y no por la ruta a la solución.

Sin embargo, resolver ingenuamente un *problema de satisfacción de restricciones*, como un problema de búsqueda regular, es enormemente ineficiente. En cambio, se puede hacer uso de la estructura de un problema de satisfacción de restricciones para resolverlo de manera más eficiente.

3.3.7 Búsqueda de retroceso

La búsqueda de retroceso es un algoritmo de búsqueda utilizado en problemas de satisfacción de restricciones. Este algoritmo se basa en una función recursiva que asigna valores a las variables del problema, respetando las restricciones establecidas. En caso de que una asignación viole alguna restricción, el algoritmo intenta una asignación diferente. A continuación, se presenta el pseudocódigo:

función Retroceder (asignación, psr):

1. Si la asignación está completa:
    - Retornar asignación.
2. var = SeleccionarVariableSinAsignar(asignación, psr).
3. Para cada valor en DominioDeValores(var, asignación, psr):
    - Si el valor es consistente con la asignación actual:
        - Agregar {var = valor} a la asignación.
        - resultado = Retroceder(asignación, psr).
        - Si resultado ≠ falla:
            - Retornar resultado.
        - Eliminar {var = valor} de la asignación.
4. Retornar falla.

El algoritmo comienza verificando si la asignación actual está completa. Si es así, retorna la asignación debido a que todas las variables han sido asignadas de forma satisfactoria. Si la asignación no está completa, selecciona una variable sin asignar y prueba cada valor posible en su dominio. Si un valor es consistente con las restricciones actuales, lo agrega a la asignación y llama recursivamente a la función Retroceder con la nueva asignación. Si esta llamada recursiva no resulta en un error, el algoritmo ha encontrado una asignación válida y la retorna. En caso de que todas las asignaciones posibles para una variable fallen, se elimina la asignación de esa variable y se retorna un

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

error, lo que induce el "retroceso" a una asignación anterior y la reconsideración de otras alternativas.

Considere el siguiente proceso de acción:

Se comienza con asignaciones vacías. Luego a la variable A (figura 3.11) se le asigna un valor, por ejemplo, Lunes. A continuación, usando esta asignación, se ejecuta el algoritmo nuevamente. Ahora que A ya tiene una asignación, el algoritmo considerará B y le asignará el Lunes. Esta asignación devuelve falso, por lo que, en lugar de asignar un valor a C dada la asignación anterior, el algoritmo intentará asignar un nuevo valor a B, por ejemplo, Martes. Esta nueva asignación satisface las restricciones y, a continuación, se considerará una nueva variable dada esta asignación.

Si, por ejemplo, asignar también Martes o Miércoles a B falla, entonces el algoritmo retrocederá y volverá a considerar A, asignándole otro valor, por ejemplo Martes. Si el martes y el miércoles también fallan, significa que se ha intentado todas las asignaciones posibles y el problema no tiene solución. El algoritmo de retroceso se usa ampliamente y, como tal, varias bibliotecas ya contienen una implementación del mismo.

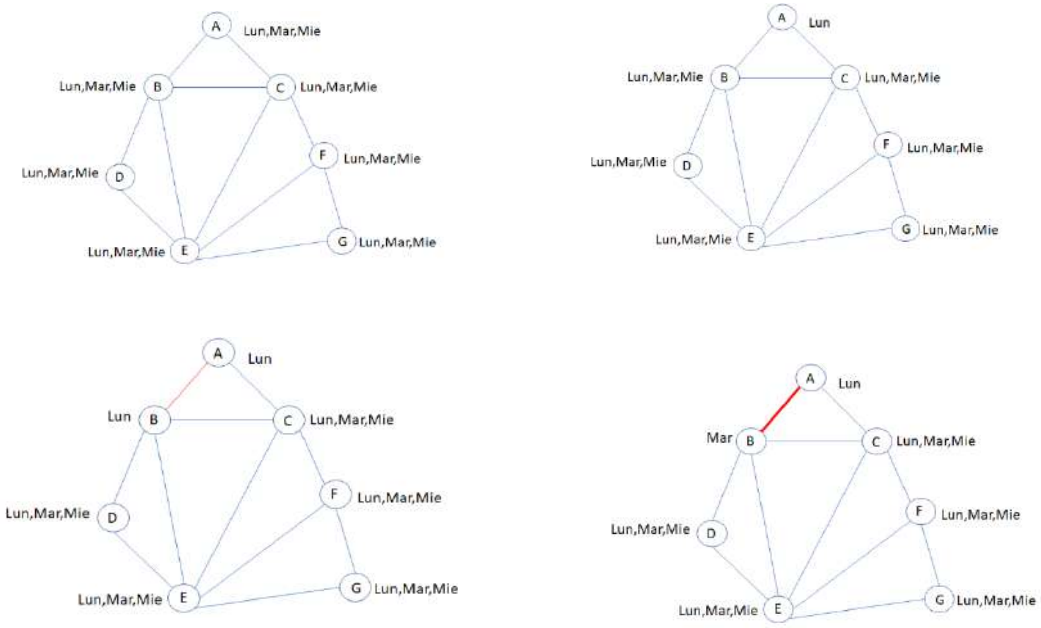


Figura 3.11 Asignación de días a las variables

Elaborado por los Autores

3.3.8 Inferencia

Aunque la búsqueda de retroceso es más eficiente que la búsqueda simple, aún requiere mucha potencia computacional. Hacer cumplir la consistencia del arco, por otro lado, requiere menos recursos. Al intercalar la búsqueda de retroceso con la inferencia

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

(reforzando la consistencia del arco), se puede tener un algoritmo más eficiente. Este algoritmo se denomina *algoritmo de mantenimiento de la consistencia del arco*. Este algoritmo hará cumplir la consistencia de arco después de cada nueva asignación de la búsqueda de retroceso. Específicamente, después de realizar una nueva asignación a X, llamaremos al algoritmo PM y lo iniciaremos con una cola de todos los arcos (Y,X) donde Y es un vecino de X (y no una cola de todos los arcos en el problema). A continuación, se muestra un algoritmo de retroceso revisado que mantiene la consistencia del arco, con las nuevas incorporaciones en negrita.

función Retroceder (asignación, psr):
- si la asignación está completa:
  - retorne la asignacción
- var = Seleccionar-Sin asignar-Var(asignación, psr)
- para valor en Dominio-Valores (var, asignación, psr):
  - si el valor es consistente con la asignación:
    - **agregar {var = valor} a la asignación**
    - **inferencias = Inferencia (asignación, psr)**
    - si inferencias ≠ falla:
      - adicionar inferencias a la asignación
    - resultado = Retroceder (asignación, psr)
    - si resultado ≠ falla:
      - retornar resultado
    - eliminar {var = valor} y inferencias de la asignación
- retornar error

La función de inferencia ejecuta el algoritmo PM como se describe. Su salida son todas las inferencias que se pueden hacer mediante la aplicación de la consistencia de arco. Literalmente, estas son las nuevas asignaciones que se pueden deducir de las asignaciones anteriores y la estructura del problema de satisfacción de restricciones.

Hay formas adicionales de hacer que el algoritmo sea más eficiente. Hasta ahora, se selecciona aleatoriamente una variable no asignada. Sin embargo, es más probable que algunas opciones lleven a una solución más rápida que otras. Esto requiere el uso de heurísticas. Una *heurística* es una regla empírica, lo que significa que, la mayoría de las veces, dará un mejor resultado que seguir un enfoque ingenuo, pero no se garantiza que así sea.

Los **valores mínimos restantes** (VMR) son una de esas heurísticas. La idea aquí es que, si el dominio de una variable se restringió por inferencia, y ahora solo le queda un valor (o incluso si son dos valores), al hacer esta asignación se reduce la cantidad de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

retrocesos que se podría necesitar hacer más adelante.

Es decir, se tendrá que hacer esta asignación tarde o temprano, debido a que se deduce de la aplicación de la consistencia de arco. Si esta tarea fracasa, es mejor averiguar lo antes posible y no retroceder más tarde.

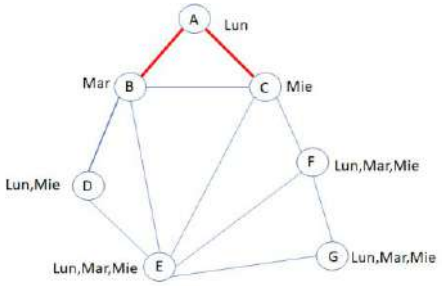


Figura 3.12 Uso de la heurística VMR

Elaborado por los Autores

Por ejemplo, después de haber acotado los dominios de las variables dada la asignación actual, utilizando la heurística VMR, elegiremos la variable C a continuación y le asignaremos el valor Miércoles (figura 3.12).

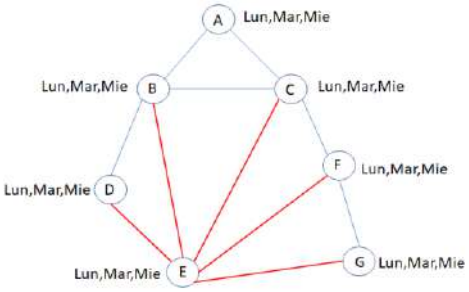


Figura 3.13 Heurística de grado

Elaborado por los Autores

La heurística de **Grado** (figura 3.13) se basa en los grados de las variables, donde un grado es cuántos arcos conectan una variable con otras variables. Al elegir la variable con el grado más alto, con una asignación, se restringen muchas otras variables, lo que acelera el proceso del algoritmo.

Por ejemplo, al analizar se debe escoger la variable E porque tiene el grado más alto de todas las variables.

Estas heurísticas no siempre son aplicables. Por ejemplo, cuando varias variables tienen el mismo número mínimo de valores en su dominio, o cuando varias variables tienen el mismo grado más alto.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Otra forma de hacer que el algoritmo sea más eficiente es emplear otra heurística cuando se selecciona un valor del dominio de una variable. Aquí, se puede usar la heurística de **Valores de Restricción Mínima**, donde seleccionamos el valor que restringirá la menor cantidad de otras variables. La idea aquí es que, mientras que en el **grado** heurístico se quería usar la variable que es más probable que restrinja otras variables, aquí queremos que esta variable imponga las menores restricciones sobre otras variables. Es decir, se quiere localizar cuál podría ser la mayor fuente potencial de problemas (la variable con el grado más alto) y luego convertirla en la menos problemática que se pueda (asignarle el valor menos restrictivo).

Por ejemplo, considérese la variable C (figura 3.14). Si se le asigna el martes, se pondrá una restricción en B, E y F. Sin embargo, si se elige el miércoles, se pondrá una restricción solo en B y E. Por lo tanto, probablemente sea mejor ir con el miércoles.

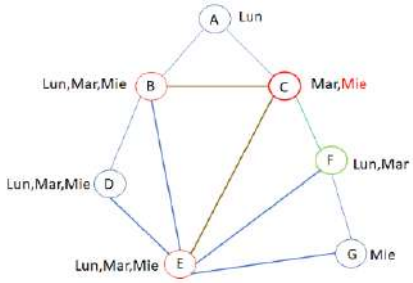


Figura 3.14 Asignación de día a la variable C

Elaborado por los Autores

3.4 Arboles de decisión

Los árboles de decisión son una de las estructuras de datos más populares y ampliamente investigadas, y son particularmente accesibles tanto para su creación como para su comprensión. Estos árboles facilitan el desarrollo de algoritmos de aprendizaje automático. Funcionan mediante la categorización de la información de entrada en función de una variable de clasificación que tiene un rango limitado de posibles valores, como "sí" o "no".

Se aplican en una variedad de áreas como en la predicción de accidentes, evaluación de la solidez crediticia, identificación de la deserción de clientes, introducción de nuevos productos en el mercado, consecución de objetivos de producción, reducción de la deserción escolar, evaluación de riesgos en proyectos, toma de decisiones legales, planificación de inversiones, previsiones climáticas, planificación de la producción,

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

diagnóstico médico, y muchos otros.

Los árboles de decisión también son útiles en sectores como el financiero y el de seguros, donde se utilizan para tomar decisiones sobre la aprobación de créditos o la determinación de las tarifas de las pólizas. Estos modelos permiten identificar qué características de los usuarios están asociadas con niveles más altos o más bajos de riesgo, facilitando la personalización en función de las necesidades y riesgos específicos de cada grupo de clientes.

Estos árboles son muy utilizados en tareas de clasificación debido a su capacidad de predecir la categoría o clase de un dato nuevo de forma eficaz, la estructura que utiliza es semejante a una serie de preguntas y respuestas que conducen a una conclusión.

Los modelos de árboles de decisión ayudan a visualizar cómo se toman las decisiones, a través de utilizar una representación que facilitan el análisis de las distintas variables involucradas. Si un nodo en el árbol contiene datos de múltiples clases, se subdivide en grupos más específicos y manejables, hasta que cada región (o nodo final) represente una sola categoría o clase.

La creación de un modelo predictivo con árboles de decisión implica varias etapas que deben llevarse a cabo de manera ordenada, como el *diseño* del modelo (selección de las características que se utilizarán en la toma de decisiones, y la construcción de la estructura del árbol), su *evaluación* (verificar qué tan bien el modelo predice o clasifica nuevos datos) y finalmente su *implementación* (integrarlo en un sistema de software existente, usarlo para tomar decisiones en tiempo real, o aplicarlo en un proceso de negocio). Para cada una de estas etapas se utiliza software o herramientas especializadas que son altamente beneficiosas, debido a que optimizan el proceso, evita errores y ahorran tiempo.

Los árboles de decisión se dividen principalmente en dos categorías: árboles de clasificación y árboles de regresión. Los árboles de clasificación se utilizan para categorizar datos en función de una variable clasificadora que es categórica, es decir, que toma un número limitado de valores posibles, como “sí” o “no”, “amarillo” o “azul”. Este tipo de árboles es útil en situaciones donde la variable objetivo es discreta y se quiere clasificar en categorías específicas. Por otro lado, los árboles de regresión se emplean cuando la variable clasificadora es continua, por lo que puede tomar cualquier valor dentro de un rango, ejemplos: peso de una persona, precio de un terreno, la temperatura

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ambiente, ventas.

Los árboles de decisión trabajan dividiendo el conjunto de datos de entrada en diferentes regiones, con el fin de que cada una de estas regiones contenga elementos que pertenezcan a una única clase. De esta forma, la clase predominante en cada región se convierte en el representante de esa región específica.

Si se encuentra una región que contiene elementos de más de una clase, el algoritmo procede a subdividirla en regiones más pequeñas y más homogéneas, siguiendo el mismo criterio de clasificación. Este proceso de subdivisión continúa hasta que todas las regiones sean lo más homogéneas posible en términos de la clase de los elementos que contienen.

Un árbol de decisión opera mediante una serie de condiciones o preguntas que se aplican a los datos de entrada. Cada condición dirige el flujo de datos hacia una rama específica del árbol, donde se realiza otra pregunta o se toma una decisión final. Este proceso se repite hasta llegar a una "hoja" del árbol, que es donde se toma la decisión final respecto a la clasificación o valor de la variable objetivo.

La "*profundidad máxima*" del árbol se refiere al número máximo de condiciones o preguntas que deben responderse para llegar a una hoja. Este es un parámetro importante porque puede afectar tanto la precisión como la complejidad del modelo. Un árbol muy profundo podría capturar detalles muy específicos de los datos de entrenamiento, pero corre el riesgo de sobre ajustar, lo que significa que podría no generalizar bien a nuevos datos. Por otro lado, un árbol muy superficial podría no capturar suficiente información, resultando en un modelo poco preciso.

Además de la consideración de la profundidad máxima del árbol, otro aspecto importante en la construcción de árboles de decisión es la "*poda del árbol*". La poda se refiere al proceso de eliminar partes del árbol, como ramas o subárboles, que no contribuyen significativamente a la capacidad de predicción del modelo. Este proceso es esencial para reducir la complejidad del modelo y evitar el sobreajuste. Mediante la poda, se pueden eliminar las ramas que se basan en variables o divisiones que no son informativas, lo que simplifica el modelo y mejora su capacidad para hacer predicciones precisas y relevantes en datos no vistos anteriormente.

Tras comprender la estructura general y la importancia de la poda en los árboles de decisión, es importante examinar los algoritmos específicos que permiten la creación y optimización de estos modelos. Existen varios algoritmos destacados en el campo de los

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

árboles de decisión, cada uno con sus características y métodos particulares. Entre los más conocidos y utilizados se encuentran el algoritmo ID3, C4.5 y CART.

Estos algoritmos difieren en cómo seleccionan las variables para dividir el árbol, cómo tratan los datos faltantes, cómo manejan variables continuas y categóricas, y en sus estrategias de poda. La elección del algoritmo adecuado depende del tipo específico de problema, la naturaleza de los datos y los objetivos del análisis.

a) ID3 (Iterative Dichotomiser 3): Desarrollado por Ross Quinlan en la década de 1980, ID3 es uno de los primeros algoritmos utilizados para construir árboles de decisión. Utiliza la Ganancia de Información como criterio para dividir los datos. ID3 se aplica principalmente a tareas de clasificación y funciona bien con atributos categóricos. Sin embargo, no es adecuado para atributos continuos y no maneja directamente los datos faltantes ni la poda de árboles, que es decisivo para evitar el sobreajuste.
b) C4.5: También desarrollado por Ross Quinlan como una extensión y mejora de ID3, C4.5 es capaz de manejar tanto datos categóricos como continuos. En lugar de la Ganancia de Información, utiliza la Razón de Ganancia de Información, que normaliza la Ganancia de Información. C4.5 introduce el concepto de poda del árbol para reducir el tamaño del árbol y mejorar su capacidad de generalización. Además, maneja los valores faltantes de los datos de manera más eficaz que ID3.
c) CART (Classification and Regression Trees): Desarrollado por Breiman, Friedman, Olshen y Stone, CART es utilizado tanto para clasificación como para regresión. A diferencia de ID3 y C4.5 que utilizan la entropía y la ganancia de información, CART utiliza el Índice Gini para clasificación y la Reducción de la Varianza para regresión. Una característica distintiva de CART es que produce árboles binarios (cada nodo tiene exactamente dos hijos), mientras que ID3 y C4.5 pueden crear árboles con más de dos hijos. CART también incluye procedimientos de poda más sofisticados para mejorar la generalización del modelo.

Una vez que hemos explorado los algoritmos clave utilizados en la construcción de árboles de decisión, es fundamental entender las métricas de división que determinan la eficacia de cada nodo en el árbol. Estas métricas, como la Ganancia de Información, el Índice Gini y la Reducción de la Varianza, son importantes en la evaluación de cómo cada característica contribuye a la separación de los datos en categorías más homogéneas. La

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

elección de la métrica adecuada puede tener un impacto significativo en el rendimiento del modelo, especialmente en la precisión con la que puede clasificar o predecir datos nuevos. Examinaremos estas métricas detalladamente para entender cómo optimizan la eficiencia y la exactitud de los árboles de decisión.

a) Ganancia de Información (Information Gain): Esta métrica se basa en el concepto de entropía de la teoría de la información. La entropía mide el grado de incertidumbre o impureza en un conjunto de datos (Poole & Mackworth, 2010). La Ganancia de Información se calcula como la diferencia en entropía antes y después de la división de un conjunto de datos. Se prefiere una división que resulte en la máxima reducción de entropía, es decir, aquella que proporciona la mayor ganancia de información. Esto significa que la división ha logrado hacer los subconjuntos de datos más homogéneos o puros respecto a la variable objetivo (Bishop & Nasrabadi, 2006).
b) Índice Gini: El Índice Gini es otra medida de impureza o variabilidad y se usa comúnmente en el algoritmo CART (Clasificación y Regresión de Árboles). Cuanto menor es el valor de Gini, mayor es la pureza del nodo. El índice Gini para un nodo se calcula sumando el cuadrado de la probabilidad de cada clase en el nodo y luego restando el resultado de uno (Géron, 2022). Cuando se considera una división, se evalúa el índice Gini para cada subconjunto resultante y se pondera por el tamaño del subconjunto. La división que resulta en el menor valor ponderado de Gini es la elegida.
c) Reducción de la Varianza: Esta métrica se utiliza principalmente para problemas de regresión en árboles de decisión. Busca minimizar la varianza de los valores objetivo en cada nodo. La idea es elegir una división tal que los valores dentro de cada subgrupo sean lo más similares posible, lo que se traduce en una baja varianza. Para cada división potencial, se calcula la varianza en cada nodo hijo y se pondera por el número de observaciones en el nodo. La división que resulta en la mayor reducción total de la varianza es seleccionada.

Por lo tanto, la profundidad máxima es un parámetro que a menudo se ajusta cuidadosamente para encontrar un equilibrio entre precisión y generalización. Este enfoque permite una clasificación efectiva y una fácil interpretación del modelo, lo que hace que los árboles de decisión sean una herramienta popular en el análisis de datos. A continuación, se tiene un ejemplo (figura 3.15).

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

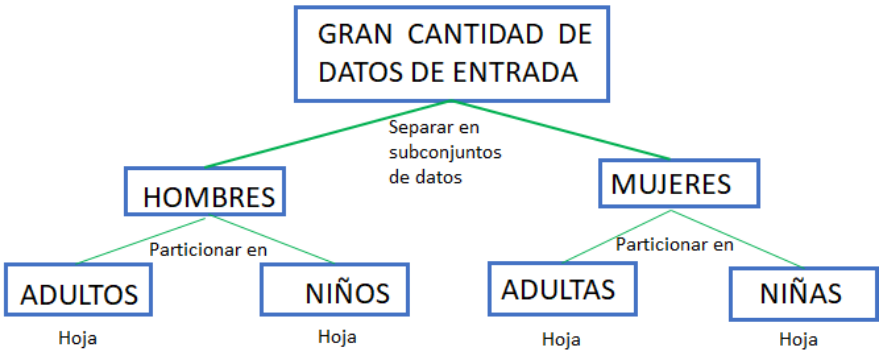


Figura 3.15 Árbol de decisión binario

Elaborado por los Autores

3.4.1 Estructura de los árboles de clasificación

Los árboles de decisión son una herramienta poderosa para tareas de clasificación y regresión. Permiten modelar decisiones complejas de una manera que es fácilmente interpretable. Existen los siguientes elementos:

Nodo Raíz: Es el punto de partida para cualquier nueva instancia de datos que se evalúa en el árbol. Este nodo contiene una condición basada en una característica o atributo del conjunto de datos, que guía el flujo de la decisión hacia uno de los nodos hijos.

El Nodo Raíz se selecciona típicamente para maximizar la diferencia o pureza entre los subconjuntos de datos resultantes. Esto se hace utilizando criterios como la ganancia de información, el índice de Gini, o la reducción de la entropía. Estos métodos buscan identificar la característica que mejor divide el conjunto de datos en grupos homogéneos en relación con la variable objetivo. Un Nodo Raíz bien elegido facilita la creación de un árbol de decisión más eficiente y preciso, debido a que cada división posterior se construye sobre esta decisión inicial.

Nodos Internos o Splits: Cada uno de estos nodos contiene una condición o pregunta basada en una característica específica del conjunto de datos. Esta condición se diseña para dividir el conjunto de datos en dos o más subconjuntos, de acuerdo con los valores que toma la característica en cuestión.

La selección de las condiciones en los Nodos Internos es un proceso estratégico, donde se busca maximizar la claridad y la distinción entre los subconjuntos resultantes. Esto se logra mediante criterios como la ganancia de información, el índice de Gini, o la entropía, que determinan qué característica y qué valor de corte generan la división más eficaz en términos de homogeneidad respecto a la variable objetivo.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Estos nodos son fundamentales en el proceso de toma de decisiones, debido a que permiten que el árbol considere múltiples características y sus interacciones. Al evaluar diferentes atributos en cada nivel del árbol, los Nodos Internos contribuyen a una mayor precisión en la clasificación o predicción, permitiendo al modelo adaptarse y responder a la complejidad inherente en los datos.

Nodos Hoja o Terminales: A diferencia de los nodos raíz y nodos internos, los nodos hoja no contienen condiciones o preguntas. En lugar de ello, proporcionan la respuesta final del árbol: asignan una etiqueta de clasificación o un valor predictivo a la instancia de datos que ha seguido el camino desde el nodo raíz, pasando por varios nodos internos, hasta llegar a este punto final.

En el contexto de la clasificación, cada nodo hoja corresponde a una categoría o clase específica. La asignación de la clase se basa en las características predominantes de las instancias de datos que llegan a este nodo, a menudo utilizando la clase mayoritaria de estas instancias como la etiqueta asignada. En los problemas de regresión, el nodo hoja típicamente representa el valor medio o mediano de las respuestas de las instancias de datos que llegan a él.

Estos nodos son fundamentales, porque encapsulan la conclusión del análisis realizado por el árbol de decisión. Proporcionan una salida clara y concreta, que es el objetivo final del proceso de modelado. La simplicidad y claridad de los nodos hoja hacen que los árboles de decisión sean herramientas altamente interpretables y transparentes en la toma de decisiones basada en datos.

Además, los nodos hoja son indicativos de la capacidad del árbol para segmentar y entender el conjunto de datos. Un equilibrio adecuado en el número y la distribución de estos nodos es importante para evitar problemas como el sobreajuste o el subajuste, asegurando así que el modelo sea generalizable y eficaz en la toma de decisiones sobre nuevos datos.

La construcción de un árbol de decisión es un proceso meticuloso que involucra varios criterios para asegurar que el modelo resultante sea tanto preciso como eficiente. A continuación, se explican los criterios:

Criterio de Parada: Este criterio es fundamental para asegurar que el modelo sea preciso y generalizable, y evitar el sobreajuste. Se establecen criterios de parada, como la profundidad máxima del árbol, un número mínimo de muestras por hoja, o un umbral de

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

ganancia de información por debajo del cual no se realizarán más divisiones.

Criterio de Selección: Se refiere a cómo se elige qué nodo (o característica) dividir en cada paso al construir el árbol, generalmente, se selecciona el nodo que ofrece la mayor ganancia de información o la mayor reducción en la impureza (como el índice de Gini o la entropía que cuantifican cuán mezcladas están las clases en un conjunto de datos) para ser dividido a continuación. Este criterio asegura que las divisiones más "informativas" se realicen primero.

Criterio de Clasificación: Una vez que se alcanza un nodo hoja, este criterio se utiliza para asignar una clase al nodo, y determinar la salida final. Esto se realiza generalmente en los árboles de clasificación, tomando la clase mayoritaria de las muestras que llegan a ese nodo; meintras que en los árboles de regresión se utiliza el promedio de los valores.

Criterio de Partición: Este criterio establece cómo se realizará la división del nodo seleccionado. En árboles binarios, cada nodo se divide generalmente en dos subnodos basados en un valor umbral de un atributo específico. Sin embargo, también existen árboles que permiten divisiones múltiples.

Estos criterios trabajan en conjunto para construir un árbol de decisión que sea tanto preciso como generalizable. Es un equilibrio delicado que a menudo requiere ajustes y validación cruzada para optimizar el rendimiento del modelo.

En la construcción de los modelos de árboles de clasificación se debe considerar:

- ✓ **Preparar los Datos**: Antes de entrenar cualquier modelo, es importante que los datos estén en un formato adecuado. Esto puede implicar la eliminación de valores nulos, vacíos, blancos. Se debe seleccionar las tablas, las columnas o variables que se desean que entren al modelo. La calidad de los datos es fundamental para el rendimiento del modelo.
- ✓ **Entrenar los Datos**: Este es el núcleo del proceso de modelado. Se utiliza un conjunto de datos de entrenamiento para ajustar el modelo. Durante esta fase, el algoritmo aprende a hacer predicciones o decisiones basadas en los datos. Los árboles de decisión, por ejemplo, aprenden a hacer divisiones basadas en los valores de las características que resultan en la mayor ganancia de información o reducción de impureza.
- ✓ **Probar el Modelo**: Una vez que el modelo ha sido entrenado, se debe probar con

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

un conjunto de datos que no haya visto antes. Esto se conoce como conjunto de datos de prueba y sirve para evaluar cómo se desempeñará el modelo en escenarios no vistos previamente.

- ✓ **Evaluar el Modelo**: La evaluación del rendimiento del modelo es crítica. Se utilizan métricas como la precisión, la sensibilidad, la especificidad y, a menudo, una matriz de confusión para entender el rendimiento del modelo. Esto ofrece una visión completa de cómo el modelo se desempeña en diferentes aspectos.
- ✓ **Aplicar el Modelo**: Una vez que se está satisfecho con el rendimiento del modelo, el último paso es desplegarlo en un entorno de producción donde pueda empezar a tomar decisiones o hacer predicciones en tiempo real.

Para garantizar la eficacia y la robustez de los modelos generados a partir de los datos existentes, es fundamental que estos modelos sean capaces de generalizarse bien a nuevos conjuntos de datos. En otras palabras, un modelo válido debe ser lo suficientemente flexible como para adaptarse a datos futuros y lo suficientemente preciso como para ser implementado en un entorno de producción.

Durante la fase de entrenamiento (figura 3.16), el modelo debe no solo ajustarse bien a los datos de entrenamiento, sino también demostrar un rendimiento sólido en datos de prueba que no haya visto anteriormente. Este equilibrio es esencial para asegurar que el modelo sea aplicable a situaciones del mundo real y no solo a las condiciones específicas de su conjunto de entrenamiento.

Es vital prevenir el fenómeno de sobreentrenamiento, que ocurre cuando un modelo se ajusta demasiado a los datos de entrenamiento y pierde su capacidad para generalizar a nuevos datos. Este problema se manifiesta cuando un modelo muestra un rendimiento excepcional en el conjunto de datos de entrenamiento, pero falla al enfrentar datos nuevos e inéditos.

Al abordar estos aspectos, se puede desarrollar un modelo de aprendizaje automático que sea tanto preciso como generalizable, lo que es fundamental para su éxito en aplicaciones prácticas.

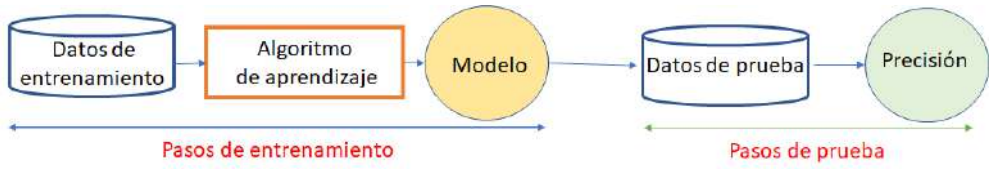


Figura 3.16 Fases de entrenamiento y prueba

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Elaborado por los Autores

Para los algoritmos supervisados es necesario contar con un conjunto de datos de entrenamiento y otro conjunto de datos de prueba, suponga que se tiene una tabla de datos de la cual el 70% se usa como datos de entrenamiento los cuales se introducen en el algoritmo de aprendizaje (árbol de clasificación) y se genera el modelo, el 30% de datos restantes son de prueba, se utiliza para aplicar el modelo y ver cuanta precisión existe con los nuevos datos.

El modelo se forma mediante repetición iterativa de entrenamiento y verificación hasta conseguir unos niveles de precisión y de capacidad de predicción aceptables.

Los conjuntos de datos de entrenamiento y de prueba se los escoge aleatoriamente de los datos iniciales. Pero existen técnicas que se utilizan para escoger la parte de datos que se quedan en el entrenamiento y en la prueba:

- Una parte para el entrenamiento y la otra parte para la prueba (70% - 30%)
- La tabla se divide en K conjuntos, se toman los K - 1 conjuntos para ir entrenado y se prueba el modelo con el último conjunto.
- Todos los datos menos 1 se utiliza para el entrenamiento y la prueba se realiza con el único dato no usado.

3.4.2 Evaluación del modelo

En el campo del aprendizaje automático, el desarrollo y entrenamiento de modelos constituyen solo una parte del proceso. Una vez que un modelo ha sido entrenado, surge una pregunta importante: ¿Qué tan bien funciona? La evaluación de modelos es un paso esencial en el ciclo de vida del aprendizaje automático, porque proporciona perspectivas críticas sobre la efectividad del modelo en la realización de predicciones o clasificaciones. Esta fase no solo es fundamental para validar la precisión del modelo, sino también para comprender sus fortalezas y debilidades, asegurando que se tomen decisiones informadas antes de su implementación en aplicaciones prácticas.

La exactitud de un modelo de aprendizaje automático rara vez es perfecta, y es aquí donde la evaluación se vuelve fundamental. Un modelo puede tener un alto grado de precisión en el conjunto de entrenamiento, pero es su rendimiento en datos no vistos anteriormente —a menudo representados por el conjunto de prueba— lo que realmente determina su utilidad. Por ello, se utilizan diversas herramientas y métricas para evaluar la calidad de un modelo, no solo en términos de su precisión general, sino también en su

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

capacidad para manejar diferentes tipos de instancias y su habilidad para evitar errores críticos.

La evaluación del modelo va más allá de la mera medición de su precisión; implica una comprensión profunda de cómo y por qué el modelo toma ciertas decisiones. Esto es especialmente importante en aplicaciones donde las consecuencias de los errores pueden ser significativas, como en el diagnóstico médico o en la toma de decisiones financieras. Por lo tanto, un modelo debe ser evaluado no solo por su capacidad para predecir correctamente, sino también por su habilidad para equilibrar diferentes tipos de errores y su adaptabilidad a diversas condiciones de datos.

Los modelos de aprendizaje automático raramente alcanzan una precisión del 100%. Por lo general, se espera que un modelo tenga un cierto nivel de exactitud, y es precisamente esta métrica la que se evalúa para determinar la eficacia del modelo.

Para llevar a cabo esta evaluación, se emplean herramientas como la *matriz de confusión*, también conocida como matriz de contingencia o matriz de errores. Esta matriz ofrece una representación visual de cómo el modelo de clasificación se desempeña al diferenciar entre las diversas clases en el conjunto de datos de prueba. No solo muestra los aciertos del modelo, sino también los tipos de errores que comete, lo que facilita una comprensión más profunda de su rendimiento.

La matriz de confusión es, en esencia, un instrumento gráfico que permite evaluar de manera cuantitativa y cualitativa el nivel de precisión de un modelo predictivo. A través de ella, se pueden calcular métricas adicionales como la sensibilidad, la especificidad y la precisión, que proporcionan una visión más completa del rendimiento del modelo (Géron, 2022).

La matriz de confusión (figura 3.17) tiene los siguientes componentes que son la clase verdadera y la clase predicha donde encontramos los *verdaderos positivos* que son aquellos ítems de los datos de la tabla que realmente llevan un elemento de verdad por ejemplo que son hombres y adultos, luego tenemos los *falsos positivos* que se predice que son hombres y adultos, pero no es verdad en realidad son hombres niños. Los *falsos negativos* son aquellos casos en los que la predicción me dice que no son hombres adultos, pero revisando la tabla si lo son, y por último los *verdaderos negativos* son aquellos casos en que se predijo que no son hombres adultos y es verdad.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

		Clase Verdadera	
		Positivos	Negativos
Clase Predicha	Positivos	Verdaderos Positivos	Falsos Positivos
	Negativos	Falsos Negativos	Verdaderos Negativos

Figura 3.17 Matriz de Confusión

Elaborado por los Autores

Verdadero Positivo (VP): Son los casos en los que el modelo predice correctamente la presencia de una característica o condición. Es el número de clasificaciones correctas en la Clase Positivos, son aquellos casos que en la clase verdadera son positivos y en la clase predicha también son positivos. Por ejemplo, si el modelo predice que ciertos individuos son hombres adultos y, efectivamente, lo son según los datos reales.

Verdadero Negativo (VN): Son los casos en los que el modelo predice correctamente la ausencia de una característica. Es el número de clasificaciones correctas en la Clase Negativos, se predijo que eran No y realmente son No. En este caso, serían los individuos que el modelo clasifica como no siendo hombres adultos, y que efectivamente no lo son según los datos reales.

Falso Negativo (FN): Son los casos en los que el modelo predice incorrectamente la ausencia de una característica. Es el número de clasificaciones incorrectas de clase positiva clasificada como negativa, el modelo predijo que eran negativos, pero realmente son positivos. Siguiendo el ejemplo, serían los individuos que el modelo clasifica como no siendo hombres adultos, pero que en realidad sí lo son.

Falso Positivo (FP): Son los casos en los que el modelo predice incorrectamente la presencia de una característica. Es el número de clasificaciones incorrectas de clase negativa clasificada como positiva, se predijo que eran positivos, pero realmente son negativos. En el ejemplo dado, serían los individuos que el modelo clasifica como hombres adultos, pero que en realidad son niños.

En el proceso de evaluación se usan las métricas las cuales proporcionan

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

información general.

Error de clasificación es la frecuencia con la que el modelo de árbol de decisión hace una predicción incorrecta. Es decir, cuántas veces el modelo clasifica incorrectamente una instancia. Este error se puede medir de varias maneras, pero generalmente se calcula como la suma de las predicciones incorrectas sobre el número total de predicciones.

$$Error = \frac{\text{FN} + \text{FP}}{\text{FN}+\text{FP}+\text{VP}+\text{VN}}$$

Exactitud (presición) es el número de predicciones correctas sobre el número total de predicciones.

$$Exactitud = \frac{\text{VP} + \text{VN}}{\text{FN}+\text{FP}+\text{VP}+\text{VN}} = 1 - Error$$

La precisión es importante, especialmente en situaciones donde los costos de los falsos positivos son altos. Se utiliza especialmente en tareas de clasificación y reconocimiento de patrones. Un ejemplo clásico de su uso es en los sistemas de filtrado de correos electrónicos para identificar y separar los correos no deseados o spam.

Recall (o Sensibilidad): También conocida como tasa de verdaderos positivos, mide la capacidad del modelo para identificar correctamente los casos positivos. Se calcula como:

$$Recall = \frac{\text{VP}}{\text{VP}+\text{FN}}$$

Es importante en contextos donde no detectar los casos positivos (como enfermedades graves) es crítico.

Especificidad: Esta métrica mide la capacidad del modelo para identificar correctamente los casos negativos. Se calcula como:

$$Especificidad = \frac{\text{VN}}{\text{VN}+\text{FP}}$$

Es importante en escenarios donde es necesario no clasificar erróneamente los negativos como positivos. Un ejemplo de uso de la métrica de especificidad es en el diagnóstico médico, donde es importante no solo identificar correctamente las condiciones presentes (sensibilidad) sino también confirmar con precisión la ausencia de una condición (especificidad).

Valor Predictivo Negativo: Mide la proporción de identificaciones negativas que fueron correctas. Se calcula como:

$$VPN = \frac{\text{VN}}{\text{VN}+\text{FN}}$$

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Continuando en elejemplo médico, esta métrica es importante en la determinación de la probabilidad de que una persona no tenga una enfermedad cuando la prueba indica un resultado negativo.

Puntuación F1: Esta es una métrica que combina la precisión y el recall en un solo número, utilizando su media armónica. Es útil cuando se busca un balance entre precisión y recall. Se calcula como:

$$\mathrm{F1} = 2\,\mathrm{x}\frac{\text{Presición x } \mathit{Recall}}{\text{Precisión} + \text{Recall}}$$

La puntuación F1 se utiliza en aplicaciones de reconocimiento de voz, como asistentes virtuales o software de transcripción, puede ser utilizada para evaluar la precisión en la identificación de palabras o frases específicas. Un alto valor de F1 indica que el sistema es eficaz tanto en reconocer correctamente las palabras (Precisión) como en capturar la totalidad de las palabras habladas (Sensibilidad).

Además de estas métricas, la ***Curva ROC*** (Receiver Operating Characteristic) y el ***Área Bajo la Curva*** (AUC) son herramientas fundamentales para evaluar la capacidad del modelo para discriminar entre clases.

La curva ROC traza la tasa de verdaderos positivos (sensibilidad) en el eje de las Y contra la tasa de falsos positivos (1 - especificidad) en el eje de las X, para diferentes umbrales de decisión, que es el punto en el que decide clasificar un resultado como positivo o negativo. Mientras que el AUC proporciona una medida agregada de rendimiento en todos los umbrales posibles. Por ejemplo, un AUC de 1 indica un modelo perfecto que clasifica todos los positivos y negativos correctamente. Un AUC de 0.5 sugiere un rendimiento no mejor que el azar. Un AUC menor a 0.5 indica un rendimiento peor que el azar, lo que podría sugerir que las etiquetas están invertidas.

ROC y AUC son herramientas fundamentales para entender qué tan bien un modelo puede diferenciar entre las clases, por ejemplo, en un modelo médico, debe distinguir correctamente entre pacientes enfermos y sanos. Estas herramientas permiten también evaluar el modelo de forma integral en un espectro de situaciones, no solo en un punto de corte fijo. Esto es útil en aplicaciones donde el equilibrio entre sensibilidad y especificidad es determinante y puede variar según el contexto.

Cada una de estas métricas aporta una dimensión diferente a la comprensión del rendimiento del modelo. Dependiendo del contexto y de los objetivos específicos del modelo, algunas métricas pueden ser más relevantes que otras. Por ejemplo, en un modelo

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

de diagnóstico médico, la sensibilidad puede ser más crítica que la precisión, mientras que, en un sistema de recomendación de productos, la precisión puede tener mayor importancia.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

BIBLIOGRAFÍA

Artasanchez, A., & Joshi, P. (2020). *Artificial Intelligence with Python: Your complete guide to building intelligent apps using Python 3. x*. Packt Publishing Ltd.

Baader, F. (2003). *The description logic handbook: Theory, implementation and applications*. Cambridge university press.

Baral, C. (2003). *Knowledge representation, reasoning and declarative problem solving*. Cambridge university press.

Barski, C. (2010). *Land of Lisp: learn to program in Lisp, one game at a time!* No starch press.

Benítez, R., Escudero, G., & Kanaan, S. (2014). *Inteligencia artificial avanzada*. Editorial UOC.

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4, Número 4). Springer.

Brachman, R., & Levesque, H. (2004a). *Knowledge representation and reasoning*. Morgan Kaufmann.

Brachman, R., & Levesque, H. (2004b). *Knowledge representation and reasoning*. Morgan Kaufmann.

Brookshear, J. G., & Brylow, D. (2020). *Computer science: an overview*. Pearson.

Chomsky, N. (2004). *Estructuras sintácticas*. Siglo xxi.

Chopra, R. (2012). *Artificial Intelligence*. S. Chand Publishing.

Colmerauer, A., & Roussel, P. (1996). *History of Programming languages—II* (T. J. Bergin Jr. & R. G. Gibson Jr., Eds.; pp. 331–367). ACM.

Croitoru, M., Marquis, P., Rudolph, S., & Stapleton, G. (2018). *Graph Structures for Knowledge Representation and Reasoning: 5th International Workshop, GKR 2017, Melbourne, VIC, Australia, August 21, 2017, Revised Selected Papers* (Vol. 10775). Springer.

Das, S. K. (2008). *Foundations of decision-making agents: Logic, Probability and Modality*. World Scientific.

Domkin, V. (2021). *Programming Algorithms in Lisp*. Springer.

Edelkamp, S., & Schrödl, S. (2011). *Heuristic search: theory and applications*. Elsevier.

Eze, T. (2022). *Trustworthy Autonomic Computing* (Vol. 30). IET.

Feigenbaum, E. A., & others. (1977). *The art of artificial intelligence: Themes and case*

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

studies of knowledge engineering.

Floridi, L. (2023). *The ethics of artificial intelligence: Principles, challenges, and opportunities*.

Franceschetti, D. R. (2018). *Principles of robotics & artificial intelligence*. Salem Press, a division of EBSCO Information Services, Incorporated.

Fraser, N. M. (1990). *Natural Language Processing in PROLOG: An Introduction to Computational Linguistics*. JSTOR.

García Serrano, A. (2012). *Inteligencia Artificial. Fundamentos, práctica y aplicaciones*. Rc Libros.

Gelfond, M., & Kahl, Y. (2014). *Knowledge representation, reasoning, and the design of intelligent agents: The answer-set programming approach*. Cambridge University Press.

Géron, A. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc."

Haykin, S. (2009). *Neural networks and learning machines, 3/E*. Pearson Education India.

Knott, G. D. (2017). *Interpreting LISP: Programming and Data Structures*. Apress.

Körner, P., Leuschel, M., Barbosa, J., Costa, V. S., Dahl, V., Hermenegildo, M. V, Morales, J. F., Wielemaker, J., Diaz, D., Abreu, S., & others. (2022). Fifty years of Prolog and beyond. *Theory and Practice of Logic Programming*, *22*(6), 776–858.

Kowaliw, T., Bredeche, N., & Doursat, R. (2014). *Growing Adaptive Machines*. Springer.

Kowalski, R. A. (2011). Artificial intelligence and human thinking. *Twenty-Second International Joint Conference on Artificial Intelligence*.

Kumar, E. (2013). *Artificial intelligence*. IK International Pvt Ltd.

Lalanda, P., McCann, J. A., & Diaconescu, A. (2013). Autonomic computing. En *Principles, Design and Implementation*. Springer.

Lee, K.-F., & Qiufan, C. (2021). *AI 2041: Ten visions for our future*. Crown Currency.

Leija, L. (2021). *Métodos de procesamiento avanzado e inteligencia artificial en sistemas sensores y biosensores*. Reverté.

Lindley, D. V. (2013). *Understanding uncertainty*. John Wiley & Sons.

MacKay, D. J. C. (2003). *Information theory, inference and learning algorithms*. Cambridge university press.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Markman, A. B. (2013). *Knowledge representation*. Psychology Press.

Martinsanz, G. P., & Peñas, M. S. (2005). *Inteligencia artificial e ingeniería del conocimiento*. Ra-Ma.

McCarthy, J. (1956). The inversion of functions defined by Turing machines. *Automata studies*, *34*, 177–181.

McCarthy, J. (1960). Recursive functions of symbolic expressions and their computation by machine, part I. *Communications of the ACM*, *3*(4), 184–195.

McCarthy, J. (1981). Epistemological problems of artificial intelligence. En *Readings in artificial intelligence* (pp. 459–465). Elsevier.

McCarthy, J. (1990). Artificial intelligence, logic, and formalising common sense. *Machine Learning and the City: Applications in Architecture and Urban Design*, 69–90.

McDermott, D. (2007). Artificial intelligence and consciousness. *The Cambridge handbook of consciousness*, 117–150.

Michiels, W., Aarts, E., & Korst, J. (2007). *Theoretical aspects of local search* (Vol. 25). Springer.

Minsky, M. (1961). Steps toward artificial intelligence. *Proceedings of the IRE*, *49*(1), 8–30.

Minsky, M., & others. (1974). *A framework for representing knowledge*. Massachusetts Institute of Technology AI Laboratory Cambridge.

Pham, D., & Karaboga, D. (2012). *Intelligent optimisation techniques: genetic algorithms, tabu search, simulated annealing and neural networks*. Springer Science & Business Media.

Poole, D. L., & Mackworth, A. K. (2010). *Artificial Intelligence: foundations of computational agents*. Cambridge University Press.

Ramirez, C. (2012). *Advances in knowledge representation*. BoD–Books on Demand.

Russell, S., & Norvig, P. (2012). Artificial intelligence—a modern approach 3rd Edition. *The Knowledge Engineering Review*. https://doi.org/10.1017/S0269888900007724

Russell, S., & Norvig, P. (2021). *Artificial intelligence: a modern approach, 4th US ed*. Pearson Education.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, *3*(3), 210–229.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

Shi, Z. (2019). *Advanced artificial intelligence* (Vol. 4). World Scientific.

Shortliffe, E. H. (1977). Mycin: A knowledge-based computer program applied to infectious diseases. *Proceedings of the Annual Symposium on Computer Application in Medical Care*, 66.

Sipser, M. (1996). Introduction to the Theory of Computation. *ACM Sigact News*, *27*(1), 27–29.

Turing, A. M. (2009). *Computing machinery and intelligence*. Springer.

Van Harmelen, F., Lifschitz, V., & Porter, B. (2008). *Handbook of knowledge representation*. Elsevier.

Weller, K. (2010). *Knowledge representation in the social semantic web*. De Gruyter Saur.

Whitby, B. (2009). *Artificial intelligence*. The Rosen Publishing Group, Inc.

Xiao, P. (2022). *Artificial intelligence programming with Python: from zero to hero*. John Wiley & Sons.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

DE LOS AUTORES

PATRICIO XAVIER MORENO VALLEJO

Ing. Patricio Xavier Moreno Vallejo M.S.

ESTUDIOS

- Master of Science Major – Computer Science and Applications, Virginia Polytechnic Institute and State University, Estados Unidos de América
- Ingeniero en Sistemas Informáticos, Escuela Superior Politécnica de Chimborazo, Ecuador

EXPERIENCIA

- Docente en la Escuela Superior Politécnica de Chimborazo desde Octubre del 2018 hasta la actualidad.
- Senior Developer en Top-Tech Advisors desde Agosto – 2018 hasta Septiembre – 2018.
- Teaching Assistant – Docente en Virginia Polytechnic Institute and State University (Virginia Tech) desde Agosto – 2016 hasta Mayo – 2017.
- Web Developer en Organization of American States (OEA) - Department of Information and Technology Services desde Enero – 2016 hasta Mayo – 2016.
- Program Developer en EasySoft S.A. desde Septiembre – 2014 hasta Septiembre – 2015.
- Desarrollador de Sistemas en Clínica Riobamba desde Mayo – 2013 hasta Mayo – 2014.
- Desarrollador Web en la Escuela Superior Politécnica de Chimborazo desde Febrero – 2013 hasta Julio – 2013.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

GISEL KATERINE BASTIDAS GUACHO

Ing. Gisel Katerine Bastidas Guacho M.S.

ESTUDIOS

- Master of Science in Computer Science, University of California, Riverside, Estados Unidos.
- Ingeniera en Sistemas Informáticos, Escuela Superior Politécnica de Chimborazo, Ecuador

EXPERIENCIA

- Docente en la Escuela Superior Politécnica de Chimborazo desde Octubre del 2019 hasta la actualidad.
- Ingeniero Senior de Business Intelligence en TopTech Advisors - Produbanco desde agosto-2018 hasta septiembre-2019.
- Analista de Tecnologías de la Información y Comunicaciones en Ministerio del Interior desde marzo-2015 hasta octubre-2015.
- Analista de TICs en CNT EP - Corporación Nacional de Telecomunicaciones desde junio-2014 hasta marzo-2015.
- Desarrolladora de Software en Clínica Riobamba desde julio-2013 hasta junio-2014.
- Desarrolladora de Software – Prácticas pre-profesionales en Escuela Superior Politécnica de Chimborazo desde febrero-2013 hasta julio-2013.
- Soporte Técnico en Scytel - CNE desde enero-2013 hasta febrero-2013.

ISBN General: 978-631-6557-23-0 ISBN Tomo 1: 978-631-6557-25-4

PATRICIO RENE MORENO COSTALES

Ing. Patricio René Moreno Costales M.S.

ESTUDIOS

- Magister en Gerencia Educativa – Universidad Estatal de Bolívar – Ecuador
- Magister en Informática Mención Redes – Universidad Andina Simón Bolívar – Escuela Politécnica Nacional - Ecuador
- Ingeniero Mecánico – Escuela Superior Politécnica de Chimborazo – Ecuador

EXPERIENCIA

- Docente Escuela Superior Politécnica de Chimborazo desde 1990
- Director Carrera de Ingeniería en Sistemas Escuela Superior Politécnica de Chimborazo 2000 a 2004
- Coordinador Carrera de Software Escuela Superior Politécnica de Chimborazo 2016 - 2023

PUERTO MADERO
EDITORIAL

ISBN 978-631-6557-23-0

ISBN 978-631-6557-25-4