Capítulo 1 – Introducción
1.1 – Qué es la IA?

- Las definiciones de IA se pueden dividir en: lo que hacen referencia a procesos mentales y razonamiento y otro a conducta.

Sistemas que piensan como humanos	Sistemas que piensan racionalmente
«El nuevo y excitante esfuerzo de hacer que los computadores piensen… máquinas con mentes, en el más amplio sentido literal». (Haugeland, 1985) «[La automatización de] actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, resolución de problemas, aprendizaje…» (Bellman, 1978)	«El estudio de las facultades mentales mediante el uso de modelos computacionales». (Charniak y McDermott, 1985) «El estudio de los cálculos que hacen posible percibir, razonar y actuar». (Winston, 1992)
Sistemas que actúan como humanos	**Sistemas que actúan racionalmente**
«El arte de desarrollar máquinas con capacidad para realizar funciones que cuando son realizadas por personas requieren de inteligencia». (Kurzweil, 1990) «El estudio de cómo lograr que los computadores realicen tareas que, por el momento, los humanos hacen mejor». (Rich y Knight, 1991)	«La Inteligencia Computacional es el estudio del diseño de agentes inteligentes». (Poole *et al.*, 1998) «IA… está relacionada con conductas inteligentes en artefactos». (Nilsson, 1998)

-

Racionalidad

- Un Sistema es racional si hace “lo correcto” en función de su conocimiento

Comportamiento humano: el enfoque de la Prueba de Turing

- La prueba de Turing fue propuesta por Alan Turing (1950)

- Diseñada para proporcionar una definición operacional y satisfactoria de inteligencia

- La prueba es medir la incapacidad de diferenciar entre entidades inteligentes indiscutibles y seres humanos

    - El computador supera la prueba si un evaluador humano no es capaz de distinguir si las respuestas a una serie de preguntas son de una persona o no

- Para que el computador supere la prueba, debe tener las siguientes capacidades:

    - **Procesamiento del lenguaje natural** que le permita comunicarse satisfactoriamente en inglés

    - **Representación del conocimiento** para almacenar lo que conoce o siente

    - **Razonamiento automático** para utilizar la información almacenada para responder a preguntas y extraer nuevas conclusiones

    - **Aprendizaje automático** para adaptarse a nuevas circunstancias y detectar y extrapolar patrones

- La prueba de Turing evita la interacción física directa entre evaluador y computador

    - La prueba global de Turing incluye una señal de video que permite al evaluador valorar la capacidad de percepción del evaluado, también se le da la oportunidad de pasar objetos físicos a través de una ventanita

    - El computador debe estar dotado de:

        - **Visión computacional:** para percibir objetos

        - **Robótica:** para manipular y mover objetos

Pensar como humano: el enfoque del modelo cognitivo

- Para poder decir que un programa piensa como humano, es necesario contar con un mecanismo para determinar cómo piensan los humanos

- Hay que penetrar en el funcionamiento de las mentes humanas. Hay dos formas:
    - Introspección: atrapar nuestros propios conocimientos conforme éstos van apareciendo
    - Experimentos psicológicos
- Cuando se cuenta con teoría lo suficientemente precisa sobre cómo trabaja la mente, se podrá expresar esa teoría en la forma de un programa
- Si los datos de entrada/salida del programa y tiempos de reacción son similares a los de un humano, existe evidencia de que algunos de los mecanismos del programa se pueden comparar con los que utilizan los seres humanos
- **Ciencia cognitiva:** convergen modelos computacional de IA y técnicas experimentales de psicología intentando elaborar teorías precisas y verificables sobre el funcionamiento de la mente humana

Pensamiento racional: el enfoque de las "leyes del pensamiento"

- **Silogismo:** esquemas de estructuras de argumentación mediante las que siempre se llega a conclusiones correctas si se parte de premisas correctas
- El estudio de estas leyes dio inicio al campo de **lógica**
- Los estudiosos de la lógica desarrollaron (XIX) una notación precisa para definir sentencias sobre todo tipo de elementos del mundo y especificar relaciones entre ellos
- La **logista** trata de construir sistemas inteligentes a partir de programas que resolvían cualquier problema resoluble descrito en notación lógica
    - Dos obstáculos

        - No es fácil transformar conocimiento informal y expresarlo en términos formales que requieren de notación lógica

        - Hay una diferencia entre poder resolver un problema “en principio” y hacerlo en la práctica

Actuar de forma racional: el enfoque del agente racional

- **Agente:** algo que razona

- Los agentes se espera que tengan otros atributos que los distingan de los “programas” convencionales: controles autónomos, perciban entorno, persistan durante un tiempo de tiempo, adapten a cambio, sean capaz de alcanzar objetivos diferentes.

- **Agente racional:** actúa con la intención de alcanzar el mejor resultado o, cuando hay incertidumbre, el mejor resultado esperado

- En IA, según las “leyes del pensamiento”, todo el énfasis es hacer inferencias correctas

    - Sin embargo efectuar una inferencia correcta no depende siempre de la racionalidad, existen situaciones para las que no hay nada correcto y hay que tomar una decisión

    - Existen también formas de actuar racionalmente que no implican realizar inferencias

        - Instintos

- Ventajas de este enfoque de diseño:
    - Es más general que el enfoque que proporcionan las “leyes del pensamiento”, dado que efectuar inferencia correctas es solo uno de los mecanismos existentes para garantizar racionalidad

    - Es más afín a la forma en la que se ha producido el avance científico que los enfoques basados en la conducta o pensamiento humano. Es de aplicación general

- Obtener la racionalidad perfecta no es posible en entornos complejos

PALABRAS CLAVE

Inteligencia	El que actúa para alcanzar el mejor resultado posible Como una materia puede entender, predecir y manipular un mundo mas grande que este
Inteligencia Artificial	Intento de entender y construir entidades inteligentes
Racionalidad	Hacer lo correcto en base a una base de conocimiento
Prueba de Turing	Prueba diseñada para proporcionar una definición operacional y satisfactoria de inteligencia La prueba es medir la incapacidad de diferenciar entre entidades inteligentes indiscutibles y seres humanos
Procesamiento de lenguaje natural	Tiene que ver con la interacción entre computadores y lenguajes humanos

	Habilidad para comunicarse en ingles
Representación del conocimiento	Habilidad para almacenar lo que el sistema conoce o siente
Razonamiento automático	Habilidad para usar la información almacenada para responder preguntas y concluir nuevas conexiones
Aprendizaje máquina (automático)	Adaptarse a nuevas circunstancias y detectar y extrapolar patrones
Prueba de Turing global	Incluye una señal de video que permite al evaluador valorar la capacidad de percepción del evaluado, también se le da la oportunidad de pasar objetos físicos a través de una ventanita Prueba de Turing diseñado para probar las capacidades físicas de visión y movimiento de un IA
Vista computacional	Percibir objetos
Robótica	Manipular y mover objetos
Ciencia cognitiva	Combina modelos computacional de IA y técnicas experimentales de psicología intentando elaborar teorías precisas y verificables sobre el funcionamiento de la mente humana
Silogismos	Esquemas de estructuras de argumentación mediante las que siempre se llega a conclusiones correctas si se parte de premisas

	correctas
Lógica	Estudio de leyes de silogismos Leyes que gobiernan la operación de la mente
Logista	Trata de construir sistemas inteligentes a partir de programas que resolvían cualquier problema resoluble descrito en notación lógica
Agente	Algo que razona
Agente racional	Actúa con la intención de alcanzar el mejor resultado o, cuando hay incertidumbre, el mejor resultado esperado
Racionalidad limitada	Actuar apropiadamente cuando no hay suficiente tiempo para hacer todas las computaciones que uno quiere

Capítulo 1 – Introducción
1.2 – Fundamentos IA

Filosofía (428 a.C – presente)

Se pueden utilizar reglas formales para extraer conclusiones válidas?

- Aristóteles (384-322 a.c.)
    - fue el primero en formar un conjunto preciso de leyes que gobernaban la parte racional de la inteligencia
    - Desarrolló un sistema informal para razonar adecuadamente con silogismos, en principio permite extraer conclusiones mecánicamente a partir de premisas iniciales
- Ramón Lull (d. 1315)
    - Tuvo la idea de que el razonamiento útil se podría obtener por medios artificiales
- Thomas Hobbes(1588-1679)
    - Propuso que el razonamiento era como la computación numérica
- Leonardo Da Vinci
    - Diseñó una calculadora mecánica
- Blaise Pascal
    - Se calculadora es la mas famosa, la máquina produce efectos que parecen mas similares a los pensamientos que a las acciones animales
- Gottfried Willhelm Leibniz
    - Construyó un dispositivo mecánico con el objetivo de llevar a cabo operaciones sobre conceptos en vez de números

Cómo se genera la inteligencia mental a partir de un cerebro físico?

- Consideremos la mente como un sistema físico:
    - René Descartes
        - Proporciona la primera discusión clara sobre la distinción entre la mente y la materia y los problemas que surgen
            - Uno de los problemas de la concepción puramente física de la mente es dejar poco margen de maniobra al libre albedrío: si el pensamiento está totalmente gobernado por leyes físicas, entonces una piedra podría decidir caer en dirección al centro de la Tierra gracias a su libre albedrío
        - Defensor del **dualismo**
            - Existe una parte en la mente (o alma o espíritu) que está al margen de la naturaleza, exenta de la influencia de las leyes físicas
            - Alternativa a dualismo es **materialismo**
                - Considera que las operaciones del cerebro realizadas de acuerdo a las leyes de la física constituyen la mente

De dónde viene el conocimiento

- Dada una mente física que gestiona el conocimiento, el siguiente problema es establecer las fuentes del conocimiento
- El movimiento **empírico**
    - Nada existe en la mente que no haya pasado antes por los sentidos
- David Hume

- Propuso lo que actualmente se le conoce como principio de **inducción:** reglas generales se obtienen mediante la exposición a asociaciones repetidas entre elementos

- Sobre esto Ludwig Wittgenstein y Bertrand Russel desarrolló la doctrina de **positivismo lógico**

    - Esta doctrina sostiene que todo el conocimiento se puede caracterizar mediante teorías lógicas relacionadas, en última instancia, con **sentencias de observación**

        - Corresponde a estímulos sensoriales

- Carnap y Carl Hempel
    - **Teoría de la confirmación**
        - Intenta explicar cómo el conocimiento se obtiene a partir de la experiencia

        - Carnap en The Logical Structure of the World define un procedimiento computacional explícito para la extracción de conocimiento a partir de experiencias primarias. Posiblemente es la prima teoría en mostrar la mente como proceso computacional

Cómo se que pasa del conocimiento a la acción

- Hay que discutir la relación que existe entre el conocimiento y la acción

- Comprender cómo se justifican determinadas acciones puede llegar a saber cómo construir un agente cuyas acciones sean justificables

- Aristóteles argumenta que las acciones se pueden justificar por la conexión lógica entre los objetivos y el conocimiento de los efectos de las acciones

- El análisis basado en objetivos es útil, pero no indica qué hacer cuando varias acciones nos llevan a la consecución del objetivo, o cuando ninguna acción facilita su completa consecución

- Antoine Arnauld
    - Describió correctamente una forma cuantitativa para decidir qué acción llevar a cabo en un caso como el anterior.
    - Propone la idea de un criterio de decisión racional en todos los ámbitos de actividad humana

---

Matemática (800-presente)

Qué reglas formales son las adecuadas para obtener conclusiones válidas?

- Los filósofos delimitaron las ideas más importante de IA, pero para pasar de ahí a una ciencia formal es necesario contar con una formulación matemática en tres áreas: lógica, computación, probabilidad

- Lógica formal
    - Remonta a los filósofos de antigua Grecia
    - Su desarrollo matemático comenzó con George Boole
        - Definió la lógica proposicional o Booleana
    - Gottlob Frege extendió la lógica de Boole para incluir objetos y relaciones, creando la lógica de primer orden que se usa hoy como el sistema más básico de representación de conocimiento
    - Alfred Tarski
        - Introdujo una toería de referencia que enseña cómo relacionar objetos de una lógica con objetos del mundo real

Qué se puede computar?

- El paso siguiente era definir los límites de lo que se podía hacer con la lógica y la informática

- El primer algoritmo no trivial fue el Euclideo para calcular el máximo común divisor

- Considerar los algoritmos como objetos en sí mismos remonta a la época de al-Khowarazmi con cuyos escritos también se introdujeron los números arábicos y álgebra

- David Hilbert
    - Presentó una lista de 23 problemas que acertadamente predijo ocuparían los matemáticos durante todo ese siglo

    - El último de ellos preguntaba si existe un algoritmo que permita determinar la validez de cualquier proposición lógica en la que aparezca números naturales (problema de decisión)

    - Lo que preguntaba es si hay límites fundamentales en la capacidad de los procedimientos efectivos de demostración

    - Kurt Gödel demostró que existe un procedimiento eficiente para demostrar cualquier aseveración verdadera en la lógica de primer orden

        - Sin embargo, en lógica de primer orden no era posible capturar el principio de inducción matemática necesario para la caracterización de los números naturales

        - Demostró que existen límites reales mediante **teorema de incompletitud**

            - Demostró que cualquier lenguaje que tuviera la capacidad suficientes para expresar las propiedades de los número naturales, existen aseveraciones verdaderas no decidible en el sentido de que no es posible decidir su validez mediante ningún algoritmo

            - Existen algunas funciones de los números enteros que no se pueden representar mediante un algoritmo, no se pueden calcular

- Esto llevó a Alan Turing a tratar de caracterizar exactamente aquellas funciones que sí eran susceptibles de ser caracterizadas

- La máquina de Turing es capaz de calcular cualquier función computable

- **Intratabilidad**
  - Un problema es intratable si el tiempo necesario para resolver casos particulares de dicho problema crece exponencialmente con el tamaño de dichos casos

  - Como se puede reconocer un problema intratable?
    - La teoría NP-Completitud propone un método

    - Los problemas clase NP-completos se puede reducir será seguramente intratable

- **Cómo razonamos con información incierta?**

- **Probabilidad**
  - Gerolamo Cardano
    - Fue el primero en proponer la idea de probabilidad, representándola en términos de los resultados de juegos de apuesta

    - Ayudó en el tratamiento de mediciones con incertidumbre y teorías incompletas

    - Pierre Fermat, Blaise Pascal, James Bernoulli, Pierre Laplace hicieron avanzar esta teoría e introdujeron nuevos métodos estadísticos

Economía (1776-presente)
Cómo se debe llevar a cabo el proceso de toma de decisiones para maximizar el rendimiento?

- Ciencia de economía comenzó en 1776 cuando el filósofo escocés Adam Smith publicó An inquirí into the Anture and Causes of the Welath of Nations

- Las economías pueden concebirse como un conjunto de agentes individuales que intentan maximizar su propio estado de bienestar económico

- Los economistas dicen que ellos realmente estudian como la gente toma decisiones que les llevan a obtener beneficios esperados

- Utilidad: beneficio deseado

- **Teoría de la decisión:** combina la teoría de la probabilidad y la de utilidad. Proporciona un marco completo y formal para la toma de decisiones realizadas bajo la incertidumbre.

- **Juego:** las acciones de un jugador pueden afectar significativamente a la utilidad de otro

- Los desarrollos a partir de la teoría de juegos mostraban el hecho de que, en algunos juegos, un agente racional debía actuar de forma aleatoria o, al menos, aleatoria en apariencia con respecto a sus contrincantes

-

Cómo se deben llevar a cabo acciones cuando otros no colaboren?

Cómo se deben llevar a cabo acciones cuando los resultados se obtienen en un futuro lejano?

- El campo de investigación operativa persigue el objetivo de tomar decisiones racionales cuando los resultados no son inmediatos y se obtienen resultados de las acciones de forma secuencial

- Richar Bellman formaliza una clase de problemas de decisión secuencial llamadas **procesos de decisión de Markov**

- El trabajo en la economía y IO ha contribuido a la noción de agente racional

- Durante muchos años la investigación en el campo de IA se ha desarrollado por sendas separadas

    - Una razón fue la **complejidad** aparente que trae consigo tomar decisiones racionales

    - Herbert Simon, uno de los primeros en investigar IA, con su temprano trabajo mostró modelos basados en **satisfacción** (toman decisiones que son suficientemente buenas, en vez de realizar cálculos laboriosos para alcanzar decisiones óptimas)

Neurociencia (1861-presente)

- La neurociencia es el estudio del sistema neurológico y en especial del cerebro

- Uno de los grandes misterios de la neurociencia es la forma exacta en la que un cerebro genera el pensamiento

- Paul Broca convenció a la sociedad médica de la existencia de áreas localizadas en el cerebro responsables de funciones cognitivas específicas. Mostró que la producción del habla se localizaba en una parte del hemisferio izquierdo

- **Neuronas:** células nerviosas que forman el cerebro

- Actualmente se dispone de información sobre la relación existente entre las áreas del cerebro y las partes del cuerpo humano que controlan o de las que perciben impulsos sensoriales

- La conclusión es que una colección de simples células puede llegar a generar razonamiento, acción y conciencia. Los cerebros generan las inteligencias.

Psicología (1879-presente)

- **Conductismo:** rechaza cualquier teoría en la que intervinieran procesos mentales, argumentado que la introspección no aportaba una evidencia fiable

    - Insistieron en el estudio exclusivo de mediciones objetivas de percepciones sobre animales y de las acciones resultantes

- **Psicología cognitiva:** conceptualización del cerebro como un dispositivo de procesamiento de información

- Kenneth Craik establece tres elementos clave que hay que tener en cuenta para diseñar un agente basado en conocimiento

    - El estímulo deberá ser traducido a una representación interna
    - Esta representación interna se debe manipular mediante procesos cognitivos para así generar nuevas representaciones internas
    - Las representaciones internas se traducirán de nuevo en acciones

- **Ciencia cognitiva:** desarrollo de modelo computacional de psicología cognitiva

    - Se pueden utilizar modelos informáticos para modelar la psicología de la memoria, el lenguaje y el pensamiento lógico

---

Ingeniería computacional (1940-presente)

- Para que IA puede llegar a se rrealidar se necesitan dos cosas:
    - Una inteligencia
    - Un artefacto

- El computador ha sido el artefacto elegido

Teoría de control y cibernética(1948-presente)

- Como pueden los artefactos operar bajo su propio control?

- La teoría matemática de los sistemas con retroalimentación estables se desarrolló en el sigo XIX

    - Norbert Weiner y equipo veía el comportamiento determinista como algo emergente de un mecanismo regulador que intenta minimizar el error

    - La teoría de control moderna tiene como objetivo el diseño de sistemas que maximizan la **función objetivo** en el tiempo.

Lingüística (1957-presente)

- Como está relacionado el lenguaje con el pensamiento?

- La teoría de Chomsky podía explicar esto. Poseía el formalismo suficiente como para permitir su programación

- La lingüística moderna y IA nacieron al mismo tiempo y maduraron juntas, volviéndose un campo híbrido llamado **lingüística computacional o procesamiento de lenguaje natural**

- El entendimiento del lenguaje requiere la comprensión de la materia bajo estudio y de su contexto

- **Representación del conocimiento:** estudio de cómo representar el conocimiento de forma que el computador pueda razonar a partir de dicha representación

PALABRAS CLAVE

Dualismo	Existe una parte en la mente (o alma o espíritu) que está al margen de la naturaleza, exenta de la influencia de las leyes físicas
Materialismo	La mente opera sobre leyes físicas
Empírico	Los sentidos son la fuente de conocimiento
Inducción	Las reglas se aprenden por medio de asociaciones repetidas reglas generales se obtienen mediante la exposición a asociaciones repetidas entre elementos
Positivismo lógico	Sostiene que todo el conocimiento se puede caracterizar mediante teorías lógicas relacionadas, en última instancia, con **sentencias de observación** Todo el conocimiento puede ser representado mediante sentencias lógicas testeables
Sentencia de observación	Estímulos sensoriales
Teoría de la confirmación	Intenta explicar cómo el conocimiento se obtiene a partir de la experiencia
Algoritmo	Una especificación de cómo solucionar un problema
Teorema de incompletitud	Demostró que cualquier lenguaje que tuviera la capacidad suficientes para expresar las propiedades de los número naturales, existen aseveraciones verdaderas no decidible en el sentido de que no

	es posible decidir su validez mediante ningún algoritmo Existen algunas funciones de los números enteros que no se pueden representar mediante un algoritmo, no se pueden calcular
Intratabilidad	Un problema es intratable si el tiempo necesario para resolver casos particulares de dicho problema crece exponencialmente con el tamaño de dichos casos
NP-Completitud	Un problema NP completo no tiene un método rápido para encontrar una solución
Probabilidad	Tiene que ver con teorías incompletas y mediciones inciertas
Teoría de la utilidad	La acción correcta depende en tus creencias y estimación de probabilidades de diferentes resultados
Teoría de la decisión	combina la teoría de la probabilidad y la de utilidad. Proporciona un marco completo y formal para la toma de decisiones realizadas bajo la incertidumbre.
Teoría de juegos	Estudio de modelos matemáticos de conflicto y cooperación entre agentes que toman decisiones inteligentes y racionales
Investigación operativa	Disciplina que utiliza modelos matemáticos, estadísticos y algoritmos para modelar y resolver problemas complejos, determinando la solución óptima y

	mejorando la toma de decisiones
Satisfacción	Tomar decisiones que son suficientemente buenas, en vez de realizar cálculos laboriosos para alcanzar decisiones óptimas
Neurociencia	El estudio del sistema neurológico y en especial del cerebro
Neuronas	células nerviosas que forman el cerebro
Conductismo	Rechaza cualquier teoría en la que intervinieran procesos mentales, argumentado que la introspección no aportaba una evidencia fiable
Psicología cognitiva	Conceptualización del cerebro como un dispositivo de procesamiento de información
Ciencia cognitiva	Junta los modelos computacionales de IA y las técnicas experimentales de psicología para construir teoría precises y testeables de la mente humana
Teoría de control	Teoría de control del comportamiento de sistemas dinámicos
Cibernética	Estudio de comunicación y control en el sistema
Función objetivo	Ecuación que será optimizada dadas las limitaciones o restricciones determinadas y con vairables que necesitan ser maximizadas o minimizadas
Lingüística computacional	La disciplina con objetivo de realizar aplicaciones informáticas que tienen la capacidad humana de hablar y entender

Capítulo 1 – Introducción
1.3 – Historia de IA

Génesis de IA (1943-1955)

- Warren McCulloch y Walter Pitts son conocidos como los autores del primer trabajo de IA
    - Partieron de:
        - Conocimiento sobre fisiología básica y funcionamiento de neuronas en el cerebro
        - Análisis formal de la lógica proposicional de Russel y Whitehead
        - Teoría de la computación de Turing
    - Propusieron un modelo constituido por neuronas artificiales en el que cada una de ellas se caracterizaba por estar “activada” o “desactivada”
        - La activación se daba como respuesta a la estimulación productiva por una cantidad suficiente de neuronas vecinas
    - Mostraron que cualquier función de cómputo podría calcularse mediante alguna red de neuronas interconectadas y que todos los conectores lógicos se podrían implementar usando estructuras de red sencillas
    - Donal Hebb propuso una sencilla regla de actualización para modificar las intensidades de las conexiones entre neuronas, su regla ahora e llama **aprendizaje Hebbiano**
- Marvin Minsky y Dean Edmonds construyeron el primer computador a partir de una red neuronal
- Turing introdujo prueba de Turing, el aprendizaje automático, algoritmos genéricos y aprendizaje por refuerzo

Nacimiento de IA (1956)

- Allen Newell y Herbert Simon contaban con un programa de razonamiento, el Teórico Lógico
    - Es un programa de computación capaz de pensar de manera no numérica

Entusiasmo inicial, grandes esperanzas (1952, 1969)

- Los investigadores de IA mostraban las computadoras realizando una tarea tras otra

- Época de “mira mama, sin manos”

- Al éxito de Newell y Simon siguió el sistema de resolución general de problemas, SRGP.

    - Este programa se diseñó para que imitara protocolos de resolución de problemas de los seres humanos

    - La secuencia en que el programa consideraba que los subobjetivos y posibles acciones eran semejantes a la manera en que los seres humanos abordan los mismos problemas

    - El éxito de SRGP llevaron a Newell y Simon formular la hipótesis de **sistema de símbolos físicos**

        - Afirma que “un sistema de símbolos físicos tiene los medios suficientes y necesarios para generar una acción inteligente”

        - En otras palabras, cualquier humano que exhibiese inteligencia debería operar manipulando estructuras de datos compuestas por símbolos

- IBM Nathaniel Rochester y colegas desarrollaron algunos de los primeros programas de IA
    - Herbert Gelernter construyó el demostrador de teoremas de geometría, era capaz de probar teoremas que muchos

estudiantes de matemáticas podían encontrar muy complejos de resolver.

- McCarthy diseño su programa para buscar la solución a problemas utilizando el conocimiento, este manejaba el conocimiento general del mundo. Generador de Consejos

    - Incorporaba los principios centrales de representación del conocimiento y el razonamiento

- **Micromundos:** dominios limitados

Una dosis de realidad (1966-1973)

- Los primeros sistemas se enfocaban en problemas simples.

- Resulto que estos primeros sistemas fallaron cuando se utilizaron en problemas mas variados o de mayor dificultad

- La mayoría de los primeros programas contaban con poco o ningún conocimiento de la materia objeto de estudio

- Muchos de los problemas que se estaban intentando resolver mediante IA eran intratables

- La ilusionaría noción de una ilimitada capacidad de cómputo no solo existió en programas para resolución de problemas

    - **Evolución automática o algoritmos genéticos**
        - Se basaba en premisa de que efectuando una adecuada serie de pequeñas mutaciones a un programa de código máquina, se podría generar un programa con buen rendimiento aplicable en cualquier tarea sencilla

    - Luego surgió la idea de probar con mutaciones aleatorias aplicando un proceso de selección con el fin de conservar aquellas mutaciones que hubiesen demostrado ser mas útiles

- El tercer problema se derivó de las limitaciones inherentes a las estructuras básicas que se utilizaban en la generación de la conducta inteligente

Sistemas basados en el conocimiento: clave del poder? (1969-1979)

- **Métodos débiles**
  - La resolución de problemas estaba centrado en el desarrollo de mecanismos de búsqueda de propósito general, en los que se entrelazaban elementos de razonamiento básicos para encontrar así soluciones completas

  - No tratan problemas más amplios o complejos

- **Marcos de Minksy**
  - Al recopilar información sobre objetos concretos y tipos de eventos, organizando estos tipos en grandes jerarquías taxonómicas similares a las biológicas

LA IA se convierte en una industria (1980-presente)

- Primer sistema experto comercial que tuvo éxito, R1

Regreso de las redes neuronales (1986-presente)

- Físicos como John Hopfield utilizaron técnicas de la mecánica estadística para analizar las propiedades de almacenamiento y optimización de redes, tratando colecciones de nodos como colecciones de átomos

- En la década de los 80 se reinventó el algoritmo de aprendizaje de retroalimentación mencionado por Bryson y Ho.

IA se convierte en una ciencia(1987-presente)

- Se ha producido una evolución tanto en el contenido como en la metodología de trabajo en el campo de IA

- Es más usual desarrollo sobre teorías ya existentes que proponer teorías totalmente novedosas

- El campo de reconocimiento del habla antes eran ad hoc, ahora con aproximaciones basadas en **modelos de Markov ocultos** han pasado a dominar el área

    - Se basan en una rigurosa teoría matemática
    - Los modelos se han generado mediante un proceso de aprendizaje en grandes corpus de datos de lenguaje reales

- Las redes neuronales se pueden comparar con otras técnicas de campos como estadística, reconocimiento de patrones y aprendizaje automático

Emergencia de los sistemas inteligentes (1995-ahora)

- Por el progreso en resolución de subproblemas de IA, los investigadores han comenzado a trabajar de nuevo en el problema del "agente total"

Conceptos importantes

Sistema de símbolos físicos	un sistema de símbolos físicos tiene los medios suficientes y necesarios para generar una acción inteligente
Micromundos	Un problema limitado que requiere inteligencia para ser resuelto
Evolución automática	Se basaba en premisa de que efectuando una adecuada serie de pequeñas mutaciones a un programa de código máquina, se podría generar un programa con buen rendimiento aplicable en cualquier tarea sencilla
Métodos débiles	Mecanismos de búsqueda de propósito general, en los que se entrelazaban elementos de razonamiento básicos para encontrar así soluciones completas
Sistemas expertos	Un programa que usa información

	disponible, heurística e inferencia para sugerir una solución a un problema de disciplina particular
Marcos	Estructura para almacenar el conocimiento Cada marco representa una clase de elementos Los hechos se organizan en una jerarquía
Conexionistas	Modelos que buscan representar fenómenos mentales usando redes neuronales artificiales
Minería de datos	Extracción automática de información útil y usualmente no conocida de bases de datos o conjuntos de datos

Capítulo 1 – Introducción
1.4 – Estado del arte

Aplicaciones de IA

- **Planificación autónoma:**
  - El agente remoto genera planes a partir de objetivos generales y monitorea las operaciones según se ejecutan los planes
- **Juegos**
- **Control autónomo**
- **Diagnosis**
  - Basado en análisis probabilístico para realizar diagnósticos
- **Planificación logística**
- **Robótica**
- **Procesamiento de lenguaje y resolución de problemas**

Capítulo 2 – Agentes inteligentes
2.1 – Agentes y su entorno

- Un agente es cualquier cosa capaz de percibir su **medioambiente** con la ayuda de **sensores** y actuar en ese medio usando **actuadores**.

- **Percepción** indica que el agente puede recibir entradas en cualquier instante

- La **secuencia de percepciones** de un agente refleja el historial completo de lo que el agente ha percibido

- Un agente tomará una decisión dependiendo de la secuencia completa de perceptores hasta ese instante

- El comportamiento del agente viene dado por la función del agente que proyecta una percepción dada en una acción

- La función que describe el comportamiento de un agente se puede presentar en forma de tabla, en la mayoría de casos sería grande

    - Dado un agente, se puede construir esta tabla teniendo en cuenta todas las secuencias de percepción y determinando qué acción lleva a cabo el agente en respuesta

- La función del agente para un agente artificia se implementará mediante el **programa del agente**

Palabras clave

Medioambiente	Fuerzas externas o actores actuando, cambiando o creando problemas
Sensores	Reciben entradas de todas las fuentes necesarias para resolver un problema
Actuadores	Controlan las acciones, producen resultados
Percepción	Las entradas que puede recibir un agente en cualquier instante
Secuencia de perceptores	El historial completo de lo que el agente ha percibido
Función del agente	Función que proyecta una percepción dada en una acción Mapeo de una percepción a una acción
Programa agente	Implementación interna de la función del agente

Capítulo 2 – Agentes inteligentes
2.2 – Buen comportamiento: el concepto de racionalidad

- Un **agente racional** es aquel que hace lo correcto

Medidas de rendimiento

- Incluyen los criterios que determinan el éxito en el comportamiento del agente

- Cuando se sitúa un agente en un medio, se genera una secuencia de acciones de acuerdo a las percepciones que recibe.

- Esta secuencia de acciones hace que su hábitat pase por una secuencia de estados

    - Si la secuencia es la deseada, entonces el agente habrá actuado correctamente

- Es mejor diseñar medidas de utilidad de acuerdo con lo que se quiere para el entorno, más de acuerdo con cómo se cree que el agente debe comportarse

Racionalidad

- La racionalidad en un momento determinado depende de cuatro factores:
    - La medida de rendimiento que define el criterio de éxito

    - El conocimiento del medio en el que habita acumulado por el agente

    - Las acciones que el agente puede llevar a cabo

    - La secuencia de percepciones del agente hasta este momento

- **Agente racional:** En cada posible secuencia de percepciones, un agente racional deberá emprender aquella acción que supuestamente maximice su medida de rendimiento, basándose en

las evidencias aportadas por la secuencia de percepciones y en el conocimiento que el agente mantiene almacenado

- Un agente puede resultar irracional en circunstancias diferentes

Omnisciencia, aprendizaje y autonomía

- Un agente omnisciente conoce el resultado de su acción y actúa de acuerdo con él, pero en realidad no es posible.

- La racionalidad maximiza el rendimiento esperado, la perfección maximiza el resultado real.

- Resulta imposible diseñar un agente que siempre lleva a cabo, de forma sucesiva, las mejores acciones después de un acontecimiento

- Llevar a cabo acciones con la intención de modificar percepciones futuras, **recopilación de información** es una parte importante de la racionalidad.

- **Exploración:** algo que lleva a cabo el agente en un medio inicialmente desconocido

- **Aprendizaje:** Conforme el agente gana experiencia, su configuración inicial puede ser modificada o aumentada.

    - La configuración inicial del agente puede reflejar un conocimiento preliminar del entorno, pero a medida que el agente adquiere experiencia éste puede modificarse y aumentar

- Los agentes con éxito dividen las taras de calcular la función del agente en tres períodos diferentes:
    - Cuando se está diseñando el agente
    - Cuando está pensando en la siguiente operación
    - Cuando está aprendiendo de la experiencia → lleva a cabo cálculos para decidir cómo modificar su forma de comportarse

- Un agente carece de **autonomía** cuando se apoya mas en el conocimiento inicial que le proporciona su diseñador que en sus propias percepciones

- Un agente racional debe ser autónomo, debe saber aprender a determinar como tiene que compensar el conocimiento incompleto o parcial inicial

Palabras clave

Agente racional	Aquel que hace lo correcto basado en su base de conocimiento
Medidas de rendimiento	Incluyen los criterios que determinan el éxito en el comportamiento del agente
Definición de agente racional	En cada posible secuencia de percepciones, un agente racional deberá emprender aquella acción que supuestamente maximice su medida de rendimiento, basándose en las evidencias aportadas por la secuencia de percepciones y en el conocimiento que el agente mantiene almacenado
Omnisciencia	Un agente omnisciente conoce el resultado de su acción y actúa de acuerdo con él
Recopilación de información	Llevar a cabo acciones con la intención de modificar percepciones futuras
Exploración	Algo que lleva a cabo el agente en un medio inicialmente desconocido
Aprendizaje	Conforme el agente gana experiencia, su configuración inicial puede ser modificada o aumentada.
Autonomía	Habilidad de saber como compensar por conocimiento parcial o incorrecto

Capítulo 2 – Agentes inteligentes
2.3 – La naturaleza del entorno

- **Entornos de trabajo:** los “problemas” para los que los agentes racionales son las soluciones

Especificación del entorno de trabajo

- **REAS (rendimiento, entorno, actuadores, sensores):** es la especificación de medidas de rendimiento, el entorno, y los actuadores y sensores del agente

- En el diseño de un agente, el primer paso debe ser siempre especificar el entorno de trabajo de la forma más completa posible

- Lo que importa no es la distinción entre un medio “real” y “rtificial” sino la complejidad de la relación entre el comportamiento del agente, la secuencia de percepción generada por el medio y la medida de rendimiento

- **Agentes software (softbots):** actúa para un usuario u otro programa

Propiedades de los entornos de trabajo

- El rango de entornos de trabajo en los que utilizan técnicas de IA es grande, pero se puede identificar un pequeño número de dimensiones en las que se puede categorizar entornos

    - Las dimensiones determinan, hasta cierto punto, el diseño mas adecuado para el agente, y la utilización de cada una de las familias principales de técnicas en la implementación del agente

- **Totalmente observable vs. Parcialmente observable**
    - Si los sensores del agente le proporcionan acceso al estado completo del medio en cada momento, entonces el entorno de trabajo es totalmente observable

- Un entorno de trabajo es totalmente observable si los sensores detectan todos los aspectos que son relevantes de la toma de decisiones

- Los entornos completamente observables son convenientes ya que el agente no necesita mantener ningún estado interno para saber qué sucede en el mundo

- Un entorno puede ser parcialmente observable debido al ruido y existencia de sensores poco exactos o porque los sensores no reciben información de parte del sistema

- **Determinista vs. Estocástico**
  - Si el siguiente estado del medio es completamente determinada por el estado actual y la acción ejecutada por el agente, entonces se dice que el entorno es determinista

    - Caso contrario, es estocástico

  - Un agente no tiene que preocuparse de la incertidumbre en un medio totalmente observable y determinista. Sin embargo si el medio es parcialmente observable entonces puede parecer estocástico

  - Si el medio es determinista, excepto para las acciones de otros agentes, decimos que el medio es **estratégico**

- **Episódico vs secuencial**
  - En un entorno de trabajo episódico, la experiencia del agente se divide en episodios atómicos

  - Cada episodio consiste en la percepción del agente y la realización de una única acción posterior

  - El siguiente episodio no depende de las acciones que se realizaron en episodios previos

  - En entornos **secuenciales**, la decisión presente puede afectar las decisiones futuras

- **Estático vs. Dinámico**
    - Si el entorno puede cambiar cuando el agente está deliberando, entonces se dice que el entorno es dinámico para el agente
        - De otra forma se dice que es estático
    - Los medios estáticos son fáciles de tratar ya que el agente no necesita estar pendiente del mundo mientras está tomando una decisión sobre una acción
    - Si el entorno no cambia con el paso del tiempo, pero el rendimiento del agente cambia entonces se dice que el medio es **semidinámico**
- **Discreto vs. Continuo**
    - La distinción entre estos se puede aplicar al estado del medio, la forma en la que se maneja el tiempo, y a las percepciones y acciones del agente
- **Agente individual vs multiagente**
    - La distinción de si un objeto se debe considerar agente o no es identificar si el comportamiento de este está mejor descrito por. La maximización de una medida de rendimiento cuyo valor depende del comportamiento de A
    - Si un agente trata maximizar su medida de rendimiento pero esto implica minimizar la medida de rendimiento de otro agente entonces tenemos un entorno multiagente **competitivo**

Palabras clave

Entornos de trabajo	Los problemas para los que los agentes racionales son las soluciones
REAS (Rendimiento, Entorno, Actuadores, Sensores)	Especificación de medidas de rendimiento, el entorno, y los actuadores y sensores del agente
Agente software	Actúa para un usuario o programa
Totalmente observable	Si los sensores del agente le proporcionan acceso al estado completo del medio en cada momento
Determinista	Si el siguiente estado del medio es completamente determinada por el estado actual y la acción ejecutada por el agente
Estocástico	Si el siguiente estado del medio no es completamente determinada por el estado actual y la acción ejecutada por el agente
Estratégico	Si el medio es determinista, excepto para las acciones de otros agentes
Episódico	En un entorno de trabajo episódico, la experiencia del agente se divide en episodios atómicos Cada episodio consiste en la percepción del agente y la realización de una única acción posterior El siguiente episodio no depende de las acciones que se realizaron

	en episodios previos
Secuencial	La decisión presente puede afectar las decisiones futuras
Estático	Si el medio puede cambiar cuando el agente está deliberando
Dinámico	El medio no puede cambiar cuando el agente está deliberando
Semidinámico	Si el entorno no cambia con el paso del tiempo, pero el rendimiento del agente cambia
Discreto	Si la acción y percepción de un agente y el estado no pueden tener un número infinito de valores en cualquier momento dado
Continuo	Si la acción y percepción de un agente y el estado pueden tener un número infinito de valores en cualquier momento dado
competitivo	Si un agente trata maximizar su medida de rendimiento pero esto implica minimizar la medida de rendimiento de otro agente
Cooperativo	Si un agente trata maximizar su medida de rendimiento pero esto implica maximizar la medida de rendimiento de otro agente

Capítulo 2 – Agentes inteligentes
2.4 – estructura de los agentes

- El **programa del agente** implementa la función del agente que proyecta las percepciones en las acciones
    - Se asume que este programa se ejecutará en algún tipo de computador con sensores físicos y actuadores
- Agente = arquitectura + programa
- El programa que se elija tiene que ser apropiado para la arquitectura
- La arquitectura hace que las percepciones de los sensores estén disponibles para el programa, ejecuta los programas, y se encarga de que los actuadores pongan en marcha las acciones generadas

Programas de los agentes

- Reciben percepciones actuales como entradas de los sensores
- Devuelven una acción a los actuadores
- Los programas de agentes toman la percepción como entrada la función de agente recibe la percepción histórica completa
- El desafío clave de IA es encontrar la forma de escribir programas que reproduzcan un comportamiento racional a partir de una pequeña cantidad de código en vez de a partir de una tabla con un gran número de entradas
- Hay cuatro tipos básicos de programas para agentes que encarnan los principios que subyacen en casi todos los sitemas inteligentes:
    - Agentes reactivos simples
    - Agentes reactivos basados en modelos
    - Agentes basados en objetivos
    - Agentes basados en utilidad

Agentes reactivos simples

- Seleccionan las acciones sobre la base de las percepciones actuales, ignorando el resto de percepciones históricas

- Poseen una inteligencia limitada

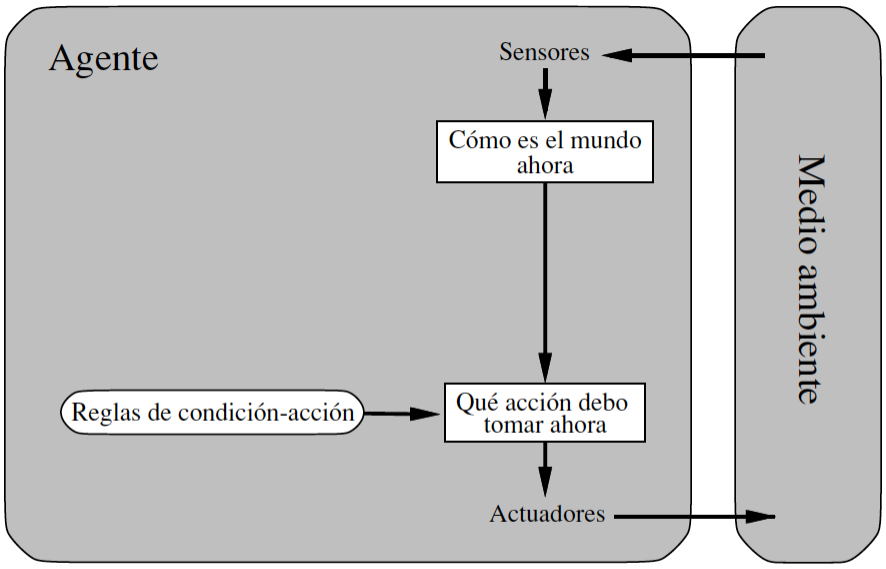


Figura 2.9 Diagrama esquemático de un agente reactivo simple.

Agentes reactivos basados en modelos

- El agente debe mantener algún tipo de **estado interno** que dependa de la historia percibida y que de ese modo refleje por lo menos alguno de los aspectos no observables del estado actual

- Mantienen un estado interno que les permite seguir el rastro de aspectos del mundo que no son evidentes según las percepciones actuales

- La actualización de información de estado interno según pasa el tiempo requiere codificar dos tipos de conocimiento en el programa agente:

    - Primero se necesita alguna información acerca de cómo evoluciona el mundo independientemente del agente

- Segundo, se necesita información sobre cómo afectan al mundo las acciones del agente

- Este conocimiento acerca de “como funciona el mundo” se le conoce como **modelo** del mundo

- Un agente que utilice este modelo es un **agente basado en modelos**

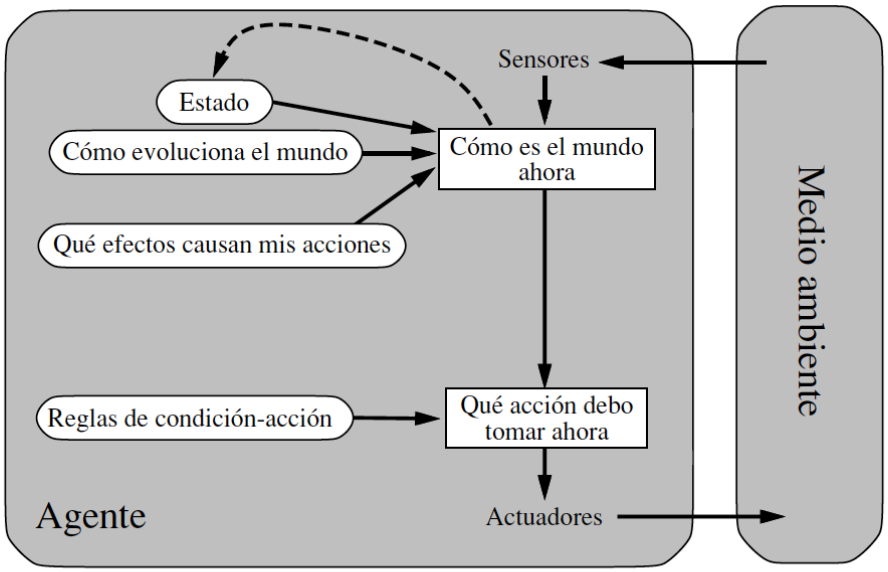


Figura 2.11 Un agente reactivo basado en modelos.

- El estado interno antiguo se combina con percepción actual para generar la descripción actualizada del estado actual.

Agentes basados en objetivos

- Además de la descripción del estado actual, el agente necesita algún tipo de información sobre su **meta**

- El programa del agente se puede combinar con información sobre los resultados de las acciones posibles (la misma información que se utilizó para actualizar el estado interno)

- En algunas ocasiones, la selección de acciones basadas en objetivos es directa, cuando alcanzar los objetivos es el resultado inmediato de una acción individual

- En otras ocasiones puede ser más complicado, cuando el agente tiene que considerar secuencias complejas para encontrar el camino que le permita alcanzar el objetivo

- La búsqueda y planificación son subcampos de IA centrados en encontrar secuencias de acciones que permitan a los agentes alcanzar sus metas.

- La toma de decisiones es diferente a las reglas condición-acción ya que hay que tener en cuenta consideraciones sobre el futuro

- Este modelo es más flexible ya que el conocimiento que soporta su decisión está representado explícitamente y puede modificarse

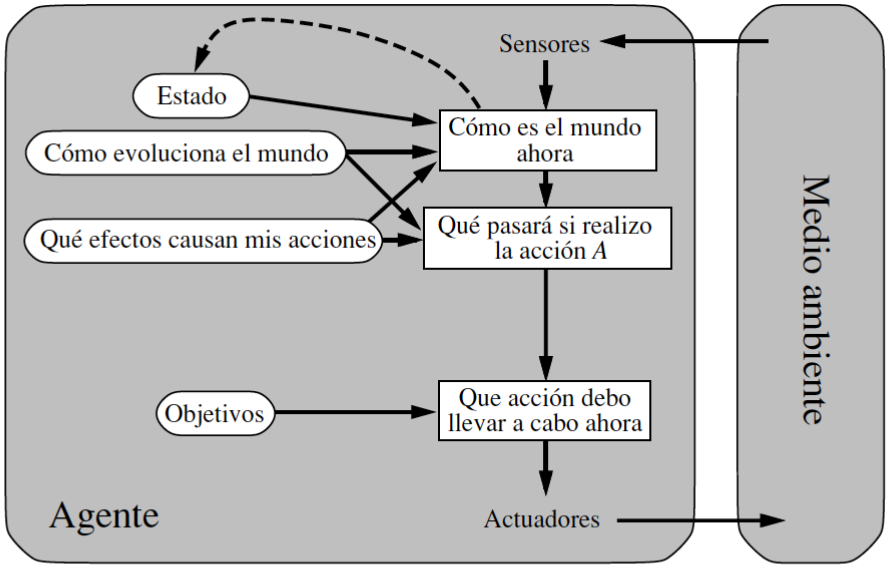


Figura 2.13 Un agente basado en objetivos y basado en modelos, que almacena información del estado del mundo así como del conjunto de objetivos que intenta alcanzar, y que es capaz de seleccionar la acción que eventualmente lo guiará hacia la consecución de sus objetivos.

Agentes basados en utilidad

- **Utilidad:** medida de “felicidad” usada para comparar entre estados del mundo diferentes que el agente alcance cuando se llegue a un estado u otro

- **Función de utilidad:** proyecta un estado (o secuencia de estados) en un número real, que representa su nivel de felicidad

- La definición completa de una función de utilidad permite tomar decisiones racionales en dos tipos de casos en los que las metas son inadecuadas:

    - Cuando haya objetivos conflictivos y solo se puedan alcanzar un de ellos (velocidad vs seguridad), la función de utilidad determina el equilibrio adecuado

    - Cuando haya varios objetivos por los que se pueda guiar al agente, y ninguno de ellos se puede alcanzar con certeza

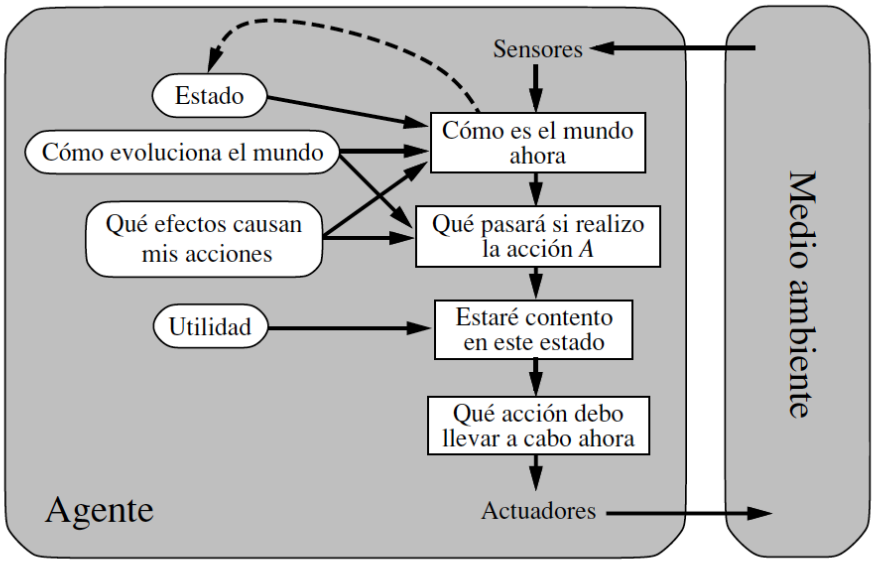


Figura 2.14 Un agente basado en utilidad y basado en modelos. Utiliza un modelo del mundo, junto con una función de utilidad que calcula sus preferencias entre los estados del mundo. Después selecciona la acción que le lleve a alcanzar la mayor utilidad esperada, que se calcula haciendo la media de todos los estados resultantes posibles, ponderado con la probabilidad del resultado.

Agentes que aprenden

- El aprendizaje permite que el agente opere en medios inicialmente desconocidos y que sea más competente que si solo utilizase un conocimiento inicial

- Un agente que aprende se puede dividir en cuatro componentes conceptuales:

    - **Elemento de aprendizaje:** está responsabilizado de hacer mejoras

    - **Elemento de actuación:** se responsabiliza de la selección de acciones externas. Recibe estímulos y determina las acciones a realizar

    - **Elemento de aprendizaje:** se realimenta con las **críticas** sobre la actuación del agente y determina cómo se debe modificar el elemento de actuación para proporcionar mejores resultados en el futuro

    - **Generador de problemas:** es responsable de sugerir acciones que lo guiarán hacia experiencias nuevas e informativas

- Todos los agentes pueden mejorar su eficacia con la ayuda de mecanismos de aprendizaje

Palabras clave

Programa del agente	implementa la función de agente que proyecta las percepciones en acciones
Arquitectura	Algún tipo de computador con sensores físicos y actuadores
Agente reactivo simple	Selecciona las acciones sobre la base de las percepciones actuales, ignorando el resto de percepciones históricas
Regla de condición-acción	La acción que el agente debe

	tomar dado una regla
Aleatorio	Resultado depende del azar
Estado interno	Información almacenada de las partes del mundo que no pueden ver
Agente basado en modelo	Un agente que utiliza conocimiento acerca de cómo funciona el mundo
Meta	Resultado esperado
Utilidad	Medida para indicar que se prefiere un estado del mundo a otro
Función de utilidad	Función que proyecta un estado (o secuencia de estados) en un número real, que representa un nivel de felicidad
Elemento de aprendizaje	está responsabilizado de hacer mejoras
Elemento de actuación	se responsabiliza de la selección de acciones externas. Recibe estímulos y determina las acciones a realizar
Crítica	Responsable para determinar cómo le está hiendo al agente y modificar el rendimiento del agente
Generador de problemas	Es responsable de sugerir acciones que lo guiarán hacia experiencias nuevas e informativas

Capítulo 3 – Resolver problemas mediante búsqueda
3.1 – Agentes resolventes-problemas

- Una clase de agente basado en objetivo se llama **agente resolvente-problemas**

    - Estos agentes deciden qué hacer al encontrar secuencias de acciones que conduzcan a estados deseables

- Los objetivos ayudan a organizar su comportamiento limitando las metas que intenta alcanzar el agente.

- La **formulación de objetivo** se basa en la situación actual y la medida de rendimiento del agente, este es el primer paso para resolver problemas

- Veamos un objetivo como un conjunto de estados del mundo (aquellos que satisfacen el objetivo).

    - La tarea del agente es encontrar qué secuencia de acciones permite obtener ese estado objetivo, para ello es necesario decidir qué acciones y estados considerar

- Dado un objetivo, la **formulación del problema** es el proceso de decidir qué acciones y estados tenemos que considerar

- Cuando un agente no sabe cual de las posibles acciones es mejor y no tiene conocimiento adicional, lo mejor que puede hacer es escoger al azar una de las acciones.

- Un agente con distintas opciones inmediatas de valores desconocidos puede decidir qué hacer, examinando las diferentes secuencias posibles de acciones que lo conduzcan a estados de valores conocidos, y entonces escoger la mejor secuencia

    - La **búsqueda** el proceso de hallar esa secuencia

- Un algoritmo de búsqueda toma como entrada un problema y devuelve una **solución** en la forma de secuencia de acciones

- Una vez encontrada una solución se ejecuta

- El diseño es formular, buscar, ejecutar

    - Después de formular un problema a resolver y objetivo, el agente llama al procedimiento de búsqueda para resolverlo. Se usa la solución para guiar sus acciones.

    - Una vez ejecutada la solución, el agente formula un nuevo objetivo

- Para resolver el problema los agentes asumen lo siguiente del entorno: estático, observable, discreto, determinístico

- En un sistema de **lazo abierto,** los agentes realizan planes con los ojos cerrados, ignoran las percepciones.

Problemas y soluciones bien definidos

- Un **problema** puede definirse por cuatro componentes:

    - **Estado inicial:** estado en el que comienza el agente

    - Una descripción de posibles **acciones** disponibles por el agente

        - La formulación más común usa una **función sucesor**

            - Dado un estado x, SUCFN(x) retorna un conjunto de pares <acción, sucesor> donde cada acción es una de las acciones legales del estado x. Cada sucesor es un estado que puede alcanzarse desde x aplicando la acción

        - El estado inicial y función sucesor forman el **espacio de estados**, todos los espacios alcanzables desde el estado inicial
            - Esto forma un grafo donde los nodos son estados y arcos son acciones

        - Un **camino** en el espacio de estados es una secuencia de estados conectados por una secuencia de acciones

    - El **test objetivo**
        - Determina si un estado es un estado objetivo

    - Una **función costo del camino**
        - Asigna costo numérico a cada camino

        - El agente resolvente de problemas elige una función costo que refleje la medida de rendimiento

        - **Costo individual:** costo de tomar la acciones “a” para ir de un estado “x” a un estado “y”. Se denota c(x, a, y)

- El problema puede definirse como una ED

- Una **solución** de problema es un camino desde el estado inicial a un estado objetivo

- La calidad de la solución se mide por la función costo del camino

- Una **solución óptima** tiene el costo más pequeño del camino entre todas las soluciones

Formular los problemas

- **Abstracción:** proceso de eliminar detalles de una representación
    - Además de abstraer la descripción del estado, debemos abstraer sus acciones
    - La abstracción es válida si podemos ampliar cualquier solución abstracta a una solución en el mundo más detallado
    - Una buena abstracción implica quitar tantos detalles como sean posibles mientras que se conserve la validez y se asegure que las acciones abstractas son fáciles de realizar

Palabras clave

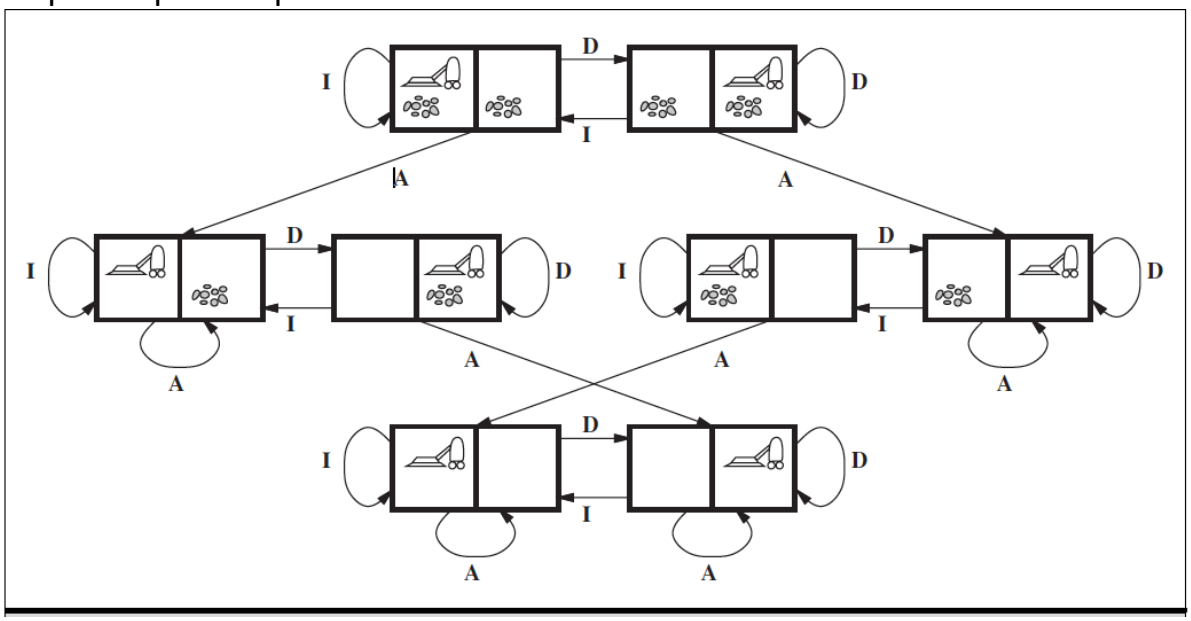
Agentes solventes-problemas	deciden qué hacer al encontrar secuencias de acciones que conduzcan a estados deseables
Formulación de objetivo	se basa en la situación actual y la medida de rendimiento del agente, este es el primer paso para resolver problemas
Formulación del problema	el proceso de decidir qué acciones y estados tenemos que considerar
Búsqueda	Examinar las diferentes secuencias posibles de acciones que lo conduzcan a estados de valores conocidos, y entonces escoger la mejor secuencia
Solución	Un camino desde el estado inicial a un estado objetivo
Ejecución	Llevar a cabo
Lazo abierto	Los agentes realizan planes con los ojos cerrados, ignoran las percepciones.
Problema	Estado inicial, descripción de posibles acciones disponibles, test objetivo y función costo del camino
Estado inicial	Estado en el que el agente empieza
Función sucesor	Una descripción de posibles acciones disponibles al agente
Espacio de estados	El estado inicial y la función sucesor lo definen. Son todos los posibles estados alcanzables desde el estado inicial
Camino	una secuencia de estados conectados por una secuencia de acciones
Test objetivo	Determina si un estado es un estado objetivo
Costo del camino	Asigna un valor numérico a cada camino

Costo individual	costo de tomar la acciones “a” para ir de un estado “x” a un estado “y”. Se denota c(x, a, y)
Solución óptima	tiene el costo más pequeño del camino entre todas las soluciones
Abstracción	Proceso de eliminar detalles de una representación

Capítulo 3 – Resolver problemas mediante búsqueda
3.2 – Ejemplos de problema

- Un **problema de juguete** se usa para ilustrar o ejercitar los métodos de resolución de problemas
    - Se puede escribir de forma exacta y concisa
- Un **problema del mundo real** es aquel en el que la gente se preocupa por sus soluciones
    - Tienden a no tener una sola descripción

Problemas de juguete

- Mundo aspiradora
    - Estados: Cada una de las dos casillas puede tener dos estados = 8
    - Estado inicial: cualquiera de los estados
    - Función sucesor: genera estados legales resultantes de aplicar izq, der, asp
    - Test objetivo: comprueba si todos los cuadrados están limpios
    - Costo de camino: cada costo individual es 1, el costo sería el numero de pasos que compone el camino
    - 

- 8 puzzle:
    - Estados: localización de cada una de las ocho fichas y el blanco en cada uno de los nueve cuadrados
    - Estado inicial: cualquier estado
    - Función sucesor: genera estados legales que resultan de aplicar las cuatro acciones (mover al blanco)
    - Test objetivo: comprueba si estado coincide con configuración objetivo
    - Costo de camino: costo de cada paso, el costo sería el número de pasos

    - Es parte de familia de puzles con piezas deslizantes

        - Se le conoce por ser NP completa

- **Formulación incremental:** implica operadores que aumentan la descripción del estado**.** El estado incial es nulo

- **Formulación completa de estados:** comienza don las descripciones del estado

Problemas del mundo real

- **Problemas de búsqueda de una ruta**
    - Estados: cada estado es una localización
    - Estado inicial: especificado por el problema
    - Función sucesor: devuelve los estados que resultan de tomar cualquier vuelo programado desde el lugar actual a otro
    - Test objetivo: tenemos nuestro destino para cierta hora especificada?
    - Costo del camino: depende del costo: dinero, tiempo de espera, tiempo de vuelo, hora, etc.

- **Problemas turísticos**
    - Relacionados a problema de búsqueda de ruta

- Aquí cada estado debe incluir no solo la localización actual sino también las ciudades que el agente ha visitado
- El objetivo seria verificar que ha recorrido todos los lugares al menos una vez

- **Problema del viajante de comercio**
  - Cada ciudad de visita exactamente una vez
  - Hay que encontrar el viaje más corto

- **Problema de distribución VLSI**
  - Colocar millones de componentes y conexiones en un chip
  - El área debe ser mínima
  - Reduce al mínimo el circuito, reduce al mínimo capacitaciones, maximiza la producción de fabricación

  - Se divide en dos partes

    - Distribución de celdas
      - Los componentes primitivos del circuito se agrupan en celdas
      - Cada celda realiza una cierta función, tiene característica fija y requiere un cierto número de conexiones a cada una de las otras celdas
      - El objetivo es colocar celdas para que quede espacio para poner alambres que conectan celdas
    - Dirección del canal
      - Encuentra una ruta especifica para cada alambre por los espacios entre celdas

- **Navegación de un robot**
  - Generalización del problema de encontrar una ruta
  - Un robot puede moverse en un espacio continuo con un conjunto infinito de acciones y estados posibles

- **Secuenciación para el ensamblaje automático**
  - Encontrar un orden en los objetos a ensamblar

    - Si se elige un orden equivocado, no habrá forma de añadir posteriormente una parte de la secuencia sin deshacer el trabajo ya hecho
      Otro problema de ensamblaje
    - **Diseño de proteínas:** el objetivo es encontrar una secuencia de aminoácidos que se plegarán en una proteína de tres dimensiones con las propiedades adecuadas para curar alguna enfermedad

- **Búsqueda en internet**: búsqueda de respuestas a preguntas, de información relacionada o para compras

Palabras clave

problema de juguete	Se usa para ilustrar o ejercitar los métodos de resolución de problemas
Problema del mundo real	Aquel en el que la gente se preocupa por sus soluciones
8-puzzle	Tablero 3x3 con 8 fichas numeradas y un espacio en blanco. El objetivo es alcanzar el acomodo de fichas especificado
Formulación incremental	implica operadores que aumentan la descripción del estado
Formulación completa de estados	comienza don las descripciones del estado
Problemas de búsqueda de una ruta	Se define en términos de localizaciones especificadas y transiciones entre ellas
Problemas turísticos	Visitar cada nodo al menos una vez

Problema del viajante de comercio	Visitar cada nodo exactamente una vez
Distribución VLSI	Colocar millones de componentes y conexiones en un chip
Navegación de un robot	Generalización del problema de encontrar una ruta
Secuenciación para el ensamblaje automático	Encontrar un orden en los objetos a ensamblar
Diseño de proteínas	El objetivo es encontrar una secuencia de aminoácidos que se plegarán en una proteína de tres dimensiones con las propiedades adecuadas para curar alguna enfermedad
Búsqueda en internet	búsqueda de respuestas a preguntas, de información relacionada o para compras

Capítulo 3 – Resolver problemas mediante búsqueda
3.3 – Búsqueda de soluciones

- Por ahora veremos las técnicas de búsqueda que utilizan un **árbol de búsqueda** explícito generado por el estado inicial y la función sucesor, definiendo el espacio de estados

- **Nodo de búsqueda:** la raíz del árbol de búsqueda
    - Es importante comprobar si este es un estado objetivo

- **Expandir** es aplicar una función sucesor al estado actual y **generar** un nuevo conjunto de estados

- La esencia de la búsqueda es llevar a cabo una opción y dejar de lado las demás para mas tarde en caso de que la escogida n conduzca una solución

- **Estrategia de búsqueda:** determina el estado a expandir

- Un nodo es una ED con los siguientes componentes:

    - ESTADO: el estado, del espacio de estados, que corresponde con nodo

    - NODO PADRE: nodo en el árbol que ha generado este nodo

    - ACCIÓN: acción que se aplicará al padre para generar el nodo

    - COSTO DEL CAMINO: costo de un camino desde estado inicial al nodo

    - PROFUNDIDAD: número de pasos a lo largo del camino desde estado inicial

- Un nodo es usado para representar el árbol de búsqueda, un estado corresponda a una configuración del mundo

- La **frontera** es una colección de nodos que se han generado pero todavía no se han expandido

- La frontera es un **nodo hoja**, no tiene sucesores
- La estrategia de búsqueda es una función que seleccione de este conjunto el siguiente nodo a expandir
    - Puede que la función estrategia tenga que mirar cada elemento del conjunto para escoger el mejor.
    - Asumiremos que la colección de nodos se implementa como una cola

Medir el rendimiento de la resolución del problema

- La salida del algoritmo de resolución es fallo o solución
- Hay cuatro formas de evaluar el rendimiento de un algoritmo
    - **Completitud:** Está garantizado que el algoritmo encuentre una solución cuando exista?
    - **Optimización:** encuentra la estrategia la solución óptima?
    - **Complejidad en tiempo:** cuánto tarda en encontrar la solución?
    - **Complejidad en espacio:** cuánta memoria se necesita para el funcionamiento de la búsqueda?
- En informática teórica, la complejidad en tiempo y espacio se considera mediante el tamaño del grafo de espacio de estados
    - En IA el grafo es representado de forma implícita y frecuentemente es infinito
        - La complejidad se expresa en términos de tres cantidades:

        - b: **factor de ramificación:** máximo número de sucesores de cualquier nodo
        - d: la profundidad del nodo objeto más superficial
        - m: la longitud máxima de cualquier camino en el espacio de estados
    - El tiempo se mide en términos de número de nodos generados durante la búsqueda
    - El espacio se mide en máximo número de nodos que se almacena en memoria

- Para valorar la eficiencia de un algoritmo de búsqueda, se puede considerar el **costo de búsqueda.** Depende de complejidad en tiempo pero puede incluir también un término para uso de memoria

- El **costo total** combina el costo de búsqueda y el costo del camino solución encontrado

Palabras clave

Árbol de búsqueda	Estructura de árbol que se usa para encontrar llaves específicas dentro de un conjunto
Nodo de búsqueda	Raíz del árbol de búsqueda
Expandir	Aplicar una función sucesor y generar un nuevo conjunto de estados
Estrategia de búsqueda	Determina el estado a expandir
Frontera	es una colección de nodos que se han generado pero todavía no se han expandido
Nodo hoja	Nodo que no tiene sucesores
Completitud	Está garantizado que el algoritmo encuentre una solución cuando exista?
Optimización	encuentra la estrategia la solución óptima?
Complejidad en tiempo	Cuanto dura en encontrar la solución
Complejidad en espacio	Cuánta memoria necesita para el funcionamiento de la búsqueda
Factor de ramificación	Máximo número de sucesores de cualquier nodo
Costo de búsqueda	Depende de complejidad en tiempo pero puede incluir también un término para uso de memoria
Costo total	combina el costo de búsqueda y el costo del camino solución encontrado

Capítulo 3 – Resolver problemas mediante búsqueda
3.4 – Estrategias de búsqueda no informada

- Una **búsqueda no informada** no tiene información adicional acerca de los estados más allá de la que proporciona la definición del problema

    - lo único que pueden hacer es generar sucesores y distinguir entre un estado objetivo de uno que no lo es

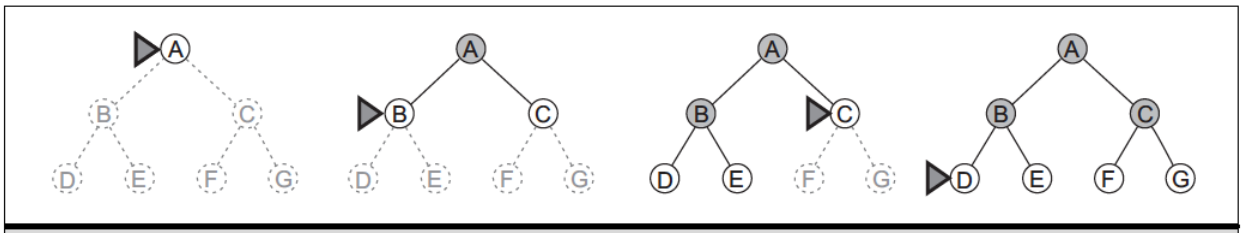
- Una **búsqueda informada** es una estrategia que sabe si un estado no objetivo es “mas prometedor” que otro

Búsqueda primero en anchura

- Estrategia sencilla en la que se expande primero el nodo raíz, luego se expanden todos los sucesores de la raíz, luego sucesores, etc.

- Se expanden todos los nodos a una profundidad antes de expandir cualquier nodo del próximo nivel

- Es completa, la complejidad en espacio es la misma que la complejidad en tiempo



Búsqueda de costo uniforme

- Se expande el nodo n con el camino de costo más pequeño

- Si todos los costos son iguales, entonces es idéntico a búsqueda primero en anchura

- No se preocupa por el número de pasos que tiene un camino, pero si el costo total

- Se puede garantizar completitud si el costo de cada paso es mayor o igual a una constante positiva pequeña, también puede garantizar optimización

- Como el costo de un camino siempre aumenta cuando vamos por él, el algoritmo expande nodos que incrementan el costo del camino, el primer nodo objetivo seleccionando para la expansión es la solución óptima

Búsqueda primero en profundidad

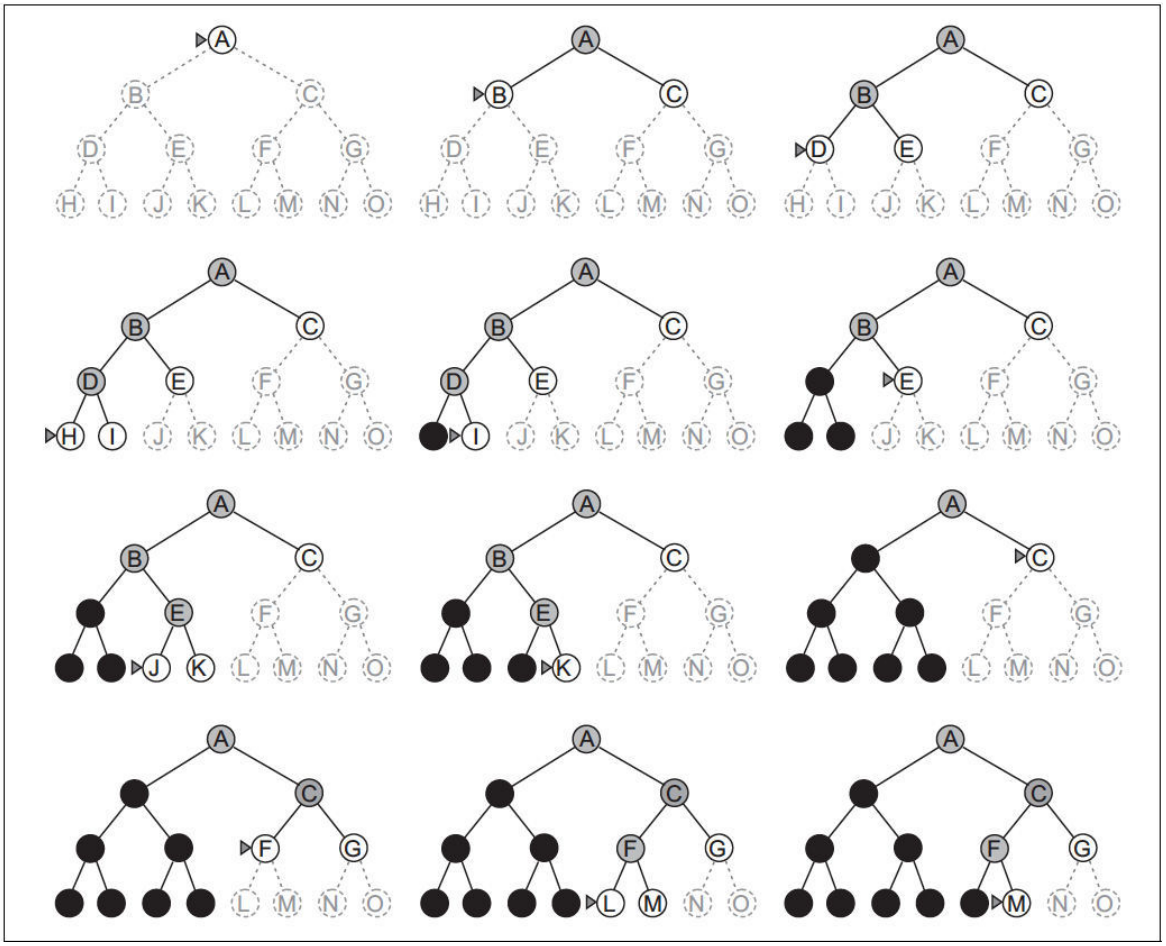
- Siempre expande el nodo más profundo en la frontera actual del árbol de búsqueda

- La búsqueda procede inmediatamente al nivel más profundo del árbol de búsqueda

- Cuando esos nodos se expanden, son quitados de la frontera, la búsqueda "retrocede" al siguiente nodo más superficial que todavía tenga sucesores inexplorados

    - Se puede implementar con cola LIFO

- Tiene requisitos modestos en memoria, necesita almacenar sólo un camino desde la raíz a un nodo hoja, junto con los nodos hermanos restantes no expandidos. Una vez el nodo se ha expandido, se puede quitar de la memoria tan pronto como todos su descendientes han sido explorados

- 

- Una variante de la búsqueda primero en profundidad es **búsqueda hacia atrás**

    - Solo se genera un nodo sucesor a la vez, cada nodo parcialmente expandido recuerda qué sucesor se expande a continuación

- Un inconveniente es que puede hacer una elección equivocada y obtener un camino muy largo aun cuando una elección diferente llevaría a una solución cerca de la raíz del árbol. No es óptima

Búsqueda de profundidad limitada

- Se hace búsqueda primero en profundidad pero se agrega un límite L de profundidad determinado. Los nodos a profundidad L se tratan como si no tuviera sucesor

- Resuelve el problema de camino infinito pero introduce problema de incompletitud si el objetivo está fuera del límite de profundidad

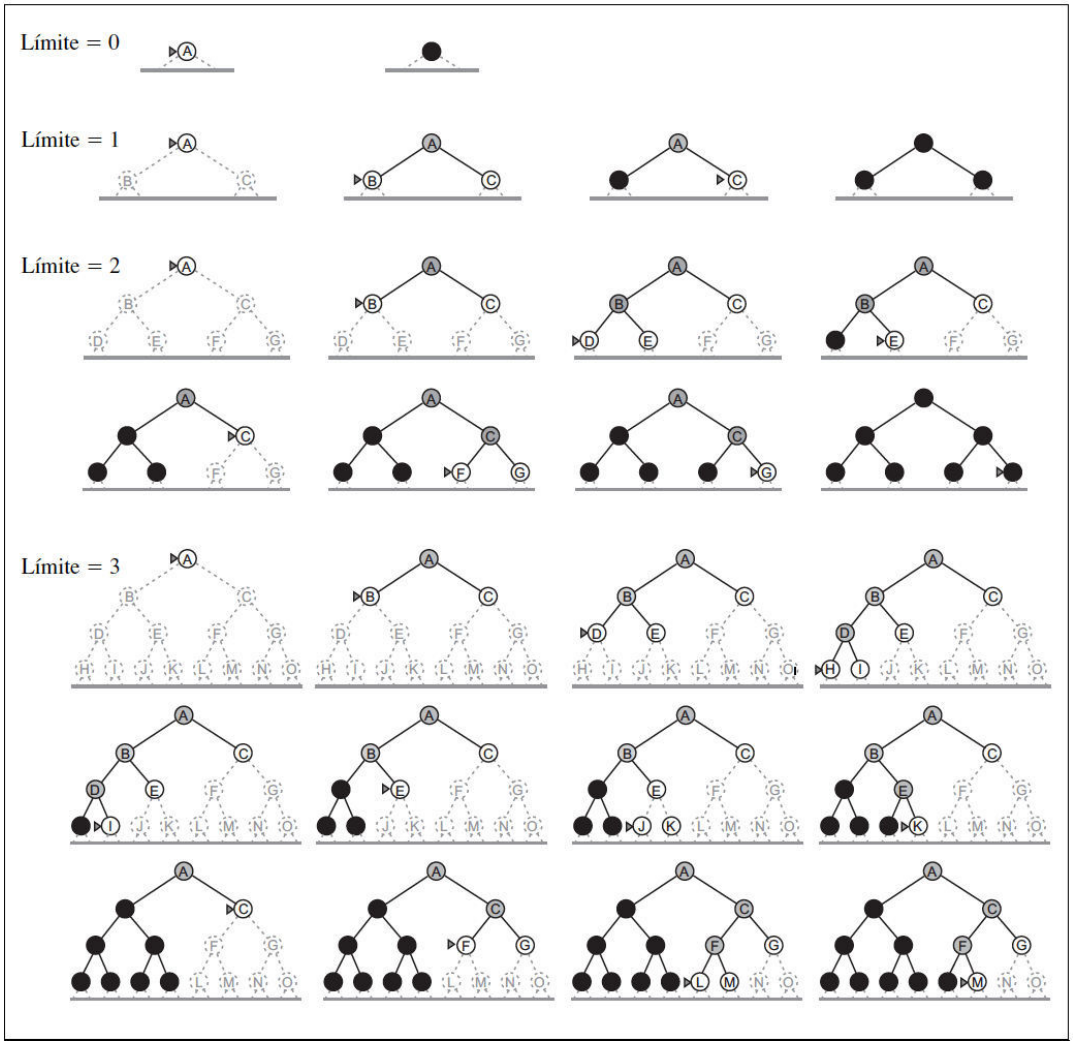
Búsqueda primero en profundidad con profundidad iterativa

- Estrategia en la cual encuentra el mejor límite profundidad

- Se usa combinada con búsqueda primero en profundidad

- Se va aumentando gradualmente el límite (0, 1, 2..) hasta encontrar un objetivo

- La profundidad iterativa es el método de búsqueda no informada preferido cuando hay un espacio grande de búsqueda y no se conoce la profundidad de la solución

- 

Búsqueda bidireccional

- Ejecutar dos búsquedas simultáneas: una hacia adelante desde el estado inicial, y la otra hacia atrás desde el objetivo

- Se implementa teniendo una o dos búsquedas que comprueban antes de ser expandido si cada nodo está en la frontera del otro árbol de búsqueda

    - Si esto ocurre, se ha encontrado una solución

- La debilidad más significativa es el espacio ya que por lo menos uno de los dos árboles de búsqueda se debe mantener en memoria

- Es completo y óptimo si las búsquedas son primero en anchura
- Como se busca hacia atrás?
    - Usando los **predecesores** todos los nodos que tienen como sucesor a “n

Comparación de las estrategias de búsqueda no informada

Criterio	Primero en anchura	Costo uniforme	Primero en profundidad	Profundidad limitada	Profundidad iterativa	Bidireccional (si aplicable)
¿Completa?	Sí[a]	Sí[a, b]	No	No	Sí[a]	Sí[a, d]
Tiempo	$O(b^{d+1})$	$O(b^{\lceil C^*/\epsilon \rceil})$	$O(b^m)$	$O(b^\ell)$	$O(b^d)$	$O(b^{d/2})$
Espacio	$O(b^{d+1})$	$O(b^{\lceil C^*/\epsilon \rceil})$	$O(bm)$	$O(b\ell)$	$O(bd)$	$O(b^{d/2})$
¿Optimal?	Sí[c]	Sí	No	No	Sí[c]	Sí[c, d]

Palabras clave

Búsqueda no informada	Estrategia que no se tiene información adicional acerca de los estados mas allá de la que proporciona la definición del problema
Búsqueda informada	Estrategia que sabe si un estado no objetivo es “mas prometedor” que otro
Búsqueda primero en anchura	Estrategia sencilla en la que se expande primero el nodo raíz, luego se expanden todos los sucesores de la raíz, luego sucesores, etc.
Búsqueda de costo uniforme	Se expande el nodo n con el camino de costo más pequeño
Búsqueda primero en profundidad	Siempre expande el nodo más profundo hasta que el nodo no tenga sucesores
Búsqueda hacia atrás	Solo se genera un nodo sucesor a la vez, cada nodo parcialmente expandido recuerda qué sucesor se expande a continuación
Búsqueda de profundidad limitada	Se hace búsqueda primero en profundidad pero se agrega un límite L de profundidad determinado. Los nodos a profundidad L se tratan como si no tuviera sucesor
Diámetro	Cantidad de pasos máximos que se toma de un nodo a otro
Búsqueda primero en profundidad con profundidad iterativa	Hace primero en profundidad pero cuando todos los nodos se han expandido y no se ha encontrado la solución, se incrementa el límite de

	profundidad
Búsqueda bidireccional	Ejecutar dos búsquedas simultáneas: una hacia adelante desde el estado inicial, y la otra hacia atrás desde el objetivo
Predecesor	Todos los nodos que tienen ese nodo como sucesor

Capítulo 3 – Resolver problemas mediante búsqueda
3.5 – Evitar estados repetidos

- Se puede perder tiempo expandiendo estados que ya han sido visitados y expandidos

- Para algunos problemas la repetición de estados es inevitable

    - Incluye problemas donde las acciones son reversibles

- Se puede cortar el árbol de búsqueda en un tamaño finito generando solo parte del árbol

- Considerando solamente el árbol de búsqueda hasta cierta profundidad fija, es fácil encontrar casos donde la eliminación de estados repetidos produce una reducción en costo de búsqueda

- La detección significa comparación del nodo a expandir con aquellos que han sido ya expandidos

- Si un algoritmo recuerda cada estado que ha visto, puede verse como la exploración directamente del grafo de espacio de datos

    - Modificamos el algoritmo general de búsqueda-arboles para incluir una estructura de datos, **lista cerrada,** almacena cada nodo ya expandido.

    - Si el nodo actual se empareja con un nodo de la lista cerrada, se elimina en vez de expandirlo

    - Puede ocurrir que eliminemos el camino recién descubierto y este era el óptimo, pero con la búsqueda de coste uniforme o búsqueda primero en anchura con costos constantes no ocurre.

Palabras clave

Rejilla rectangular	Sobre una rejilla, cada estado tiene cuatro sucesores
Lista cerrada	Almacena cada nodo expandido

Capítulo 3 – Resolver problemas mediante búsqueda
3.6 – Búsqueda con información parcial

- Hasta este punto habíamos asumido que el entorno era observable, determinista y el agente conoce cuáles son los efectos de cada acción

- Diversos tipos de incompletitud conducen a tres tipos de problemas distintos:

    - **Problemas sin sensores:**
        - Si el agente no tiene ningún sensor entonces podría estar en uno de los posibles estados iniciales, y cada acción podría conducir a uno de los posibles estados sucesores

    - **Problemas de contingencia:**
        - Si el entorno es parcialmente observable, o si las acciones son inciertas, las percepciones del agente proporcionan nueva información después de cada acción

        - Cada posible percepción define nueva información después de cada acción

        - Cada posible percepción define una contingencia la cual se debe planear

            - A un problema se le dice entre **adversarios** si la incertidumbre está causado por las acciones de otro agente

    - **Problemas de exploración**
        - Cuando se desconocen los estados y acciones del entorno, el agente debe actuar para descubrirlos

Problemas sin sensores

- El agente puede **coaccionar** al mundo

- Cuando el mundo no es completamente observable, el agente debe decidir sobre los conjuntos de estados que podría poner

    - Cada conjunto de estados es un estado de creencia ya que el agente actual cree en los estados físicos posibles que podría estar

- Para resolver problemas sin sensores, buscamos en el espacio de estados de creencia más que en los estados físicos

    - El estado inicial es un estado de creencia, y cada acción aplica un estado de creencia a otro de creencia.

    - Una acción se aplica a un estado de creencia uniendo todos los resultados de aplicar la acción a cada estado físico del estado de creencia

    - Un camino une varios estados de creencia y una solución es ahora un camino que conduce a un estado de creencia, todos de cuyos miembros son estados objetivo

Problemas de contingencia

- Cuando el entorno es tal que el agente puede obtener nueva información de sus sensores después de su actuación, el agente afronta **problemas de contingencia**

- La solución toma forma de árbol, donde cada rama puede seleccionar según la percepción percibida en ese punto del árbol

- Se puede ampliar el espacio de soluciones para incluir la posibilidad de seleccionar acciones basados en contingencias durante la ejecución

- Los problemas de contingencia a veces permiten soluciones puramente secuenciales

- Los algoritmos para problemas de contingencia son más complejos que los algoritmos estándar de búsqueda

- los problemas de contingencia también se prestan a un diseño de agente diferente, donde el agente puede actuar antes de que haya encontrado un plan garantizado

Palabras clave

Coacción	forzar
Estados de creencia	La creencia del agente con los estados posibles físicos a los que podría llegar
Problemas de contingencia	Cuando el entorno es tal que el agente puede obtener nueva información de sus sensores después de su actuación

4 - Búsqueda informada y exploración
4.1 – Estrategias de búsqueda informada (heurísticas)

- La estrategia de búsqueda informada usa conocimiento específico sobre el problema

- La aproximación general es **búsqueda primero el mejor**

    - Es un caso particular del algoritmo general de búsqueda de árboles o grafos

    - Se selecciona un nodo para la expansión basada en una **función de evaluación**

        - Mide la distancia al objetivo, f.

    - Se puede implementar con una cola de prioridad, que mantendrá la frontera en orden ascendente de valores f.

- Hay una familia entera de algoritmos de búsqueda primero mejor con funciones de evaluación diferentes

    - Un componente clave de estos algoritmos es una **función heurística** h(n) = costo estimado del camino más barato desde nodo n a un nodo objetivo

Modos para de usar la información heurística para dirigir la búsqueda

Búsqueda voraz primero el mejor

- Trata de expandir el nodo más cercano al objetivo

- Se evalúan los nodos usando solamente la función heurística: f(n) = h(n)

- Es algoritmo avaro porque en cada paso trata de ponerse tan cerca del objetivo como pueda

- Si no somos cuidados en descubrir estados repetidos, la solución puede que nunca se encontrará

- Se parece a la búsqueda primero en profundidad en el modo que prefiere seguir un camino hacia el objetivo, pero volverá atrás cuando llegue a un callejón sin salida

- No es óptima y es incompleta (puede irse hasta abajo y no probar las otras opciones)

Búsqueda A*: minimizar el costo estimado total de la solución

- Evalúa los nodos combinando g(n), el costo para alcanzar el nodo, y h(n), el costo de ir al nodo objetivo

    - f(n) = g(n) + h(n)

    - g(n) nos da el costo del camino desde el nodo inicio al nodo "n" y h(n) el costo estimado del camino más barato desde "n" al objetivo:

        - f(n) = costo más barato estimado de la solución a través de n

        - Es razonable intentar primero el nodo con el valor más bajo de g(n) + h(n)

        - Resulta más razonable que con tal de que la función heurística h(n) satisfaga ciertas condiciones, la búsqueda A* es tanto completa como óptima

- A* es óptima si h(n) es una **heurística admisible,** con tal de que h(n) nunca sobrestime el costo de alcanzar el objetivo

    - Son optimistas por naturaleza, piensan que el costo de resolver el problema es menor que es en realidad

- **Consistencia:** una heurística es consistente si para todo nodo “n” y cada sucesor “n’ “ de n, generado por cualquier acción a, el costo estimado de alcanzar el objetivo desde n no es mayor que el costo de alcanzar n’ más el costo estimado de alcanzar el objetivo desde n’

    - $H(n) <= c(n, a, n') + h(n')$

        - Esto es una forma de **desigualdad triangular**
            - Especifica que cada lado de un triángulo no puede ser más largo que la suma de los otros dos lados

            - En nuestro caso el triangulo está formado por n, n’ y el objetivo mas cercano a n

- El hecho de que los f-costos no disminuyan a lo largo de cualquier camino significa que se pueden dibujar **curvas de nivel** en el espacio de estados
    - Dentro de la curva de nivel, todos los nodos tienen f(n) menor que ese valor

- El algoritmo A* es **óptimamente eficiente** para cualquier función heurística, ningún otro algoritmo garantiza expandir menos nodos que A*

Búsqueda heurística con memoria acotada

- **Búsqueda recursiva del primero mejor (BRPM)**
    - Algoritmo recursivo que intenta imitar la operación de búsqueda primero el mejor, pero utiliza solo un espacio lineal

    - Estructura similar a la búsqueda primero en profundidad pero es seguir indefinidamente hacia abajo en el camino actual

    - Mantiene la pista del f-valor del mejor camino alternativo disponible desde cualquier antepasado del nodo actual

- Si el nodo actual excede este límite, la recursividad vuelve atrás al camino alternativo

- BRPM sustituye los f-valores de cada nodo a lo largo del camino actual

- Mantiene la pista del f-valor del mejor camino alternativo disponible desde cualquier antepasado del nodo actual

- Si el nodo actual excede el límite, la recursividad vuelve al camino alternativo , BRPM sustituye los f-valores de cada nodo a lo largo del camino con el mejor f-valor de su hijo

- Es un algoritmo óptimo y la función heurística h(n) es admisible

- A*PI y BRPM sufran de utilizar muy poca memoria

- Parece sensible usar toda la memoria disponible
  - Dos algoritmo que hacen esto son A*M y A*MS, memoria acotada y memoria simplificada respectivamente

  - A*MS

    - Avanza como A*, expandiendo la mejor hoja hasta que la memoria esté llena

    - Para retirar, el algoritmo retira el peor nodo hoja (f-valor mas alto)

    - Se devuelve hacia atrás, a su padre

    - El antepasado de un subárbol olvidado sabe la calidad del mejor camino en el subárbol, A*MS vuelve a generar el subárbol solo cuando todos los otros caminos parecen peores que el camino olvidado

Aprender a buscar mejor

- Podría un agente aprender a buscar mejor?

- Si, el método se poya sobre un concepto llamado **espacio de estados metanivel**

    - Cada espacio captura el estado interno de un programa que busca en un **espacio de estados a nivel de objeto**

    - Cada acción en el espacio de estados es un paso de cómputo que cambia el estado interno

Palabras clave

Búsqueda informada	Usa conocimiento específico sobre el problema más allá de la definición del problema en sí mismo
Búsqueda primero el mejor	Se selecciona un nodo para la expansión basada en una función de evaluación
Función de evaluación	Mide la distancia al objetivo
Función heurística	costo estimado del camino más barato desde nodo n a un nodo objetivo
Búsqueda voraz primero el mejor	Trata de expandir el nodo más cercano al objetivo
Búsqueda A*	Evalúa los nodos combinando g(n), el costo para alcanzar el nodo, y h(n), el costo de ir al nodo objetivo
Heurística admisible	Son optimistas por naturaleza, piensan que el costo de resolver el problema es menor que es en realidad
óptimamente eficiente	Ningún otro algoritmo óptimo garantiza expandir menos nodos
Búsqueda recursiva del primero mejor	

4 - Búsqueda informada y exploración
4.2 – Funciones heurísticas

- Una manera de caracterizar la calidad de una heurística es b*, el factor de ramificación eficaz

- b* es el factor de ramificación que un árbol uniforme de profundidad d debería tener para contener N + 1 nodos

- Una heurística bien diseñada tendría un valor b* cerca de 1

Inventar funciones heurísticas admisibles

- Un **problema relajado** es un problema con menos restricciones en las acciones

- El costo de una solución óptima en un problema relajado es una heurística admisible para el problema original

- También se pueden obtener heurísticas admisibles del costo de la solución de un sub problema.

- La idea del **modelo de base de datos** es almacenar costos exactos de las soluciones para cada posible sub problema.

Aprendizaje de heurísticas desde la experiencia

- Usar experiencia, resolver muchas veces un problema

- A partir de las corridas se puede crear un **algoritmo de aprendizaje inductivo** para construir una función h(n) que pueda predecir todos los costos solución para otros estados que surjan durante la búsqueda

- Los métodos de aprendizaje inductivo trabajan mejor cuando se les suministran características de un estado que sean relevantes para su evaluación

4 - Búsqueda informada y exploración
4.3 – Algoritmos de búsqueda local y problemas de optimización

- Tenemos problemas donde el camino al objetivo es irrelevante

- Los algoritmos de **búsqueda local** funcionan con un solo **estado actual** y generalmente se mueve solo a los vecinos del estado

- Los caminos seguidos no se retienen

- Son útiles para resolver problemas de optimización puros, en los que e, objetivo es encontrar el mejor estado según la función objetivo

- El paisaje de espacio de datos tiene:

    - Posición (definido por el estado)
    - Elevación (definido por el valor de función de costo de heurística o función objetivo)

    - Si la elevación es un costo, el objetivo es encontrar el valle mas bajo, mínimo global.

Búsqueda en ascensión de colinas

- Es un bucle que continuamente se mueve en dirección del valor creciente

- Termina cuando alcanza un pico en donde ningún vecino tiene un valor más alto

- Los algoritmos de búsqueda local típicamente usan una **formulación de estados completa**

- Destaca por:

    - **Máximo local**
    - **Cresta:** crean una secuencia de máximos locales que hace muy difícil la navegación para los algoritmos avaros

    - **Meseta:** área del paisaje de espacio de estados donde la función de evaluación es plana

- El éxito de la ascensión de colinas depende muchísimo de la forma del paisaje del espacio de estados

Búsqueda de temple simulado

- Un algoritmo de ascensión de colinas que nunca hace movimientos hacia estados con un valor inferior garantiza ser incompleto porque puede estancarse en un máximo local

- Parece razonable intentar combinar la ascensión de colinas con un camino aleatorio de algún modo que produzca tanto eficacia como completitud, el **temple simulado** papus.

    - Hay que cambiar el punto de vista a **gradiente descendente**

    - El truco es sacudir con bastante fuerza para echar la pelota de mínimos locales no lo suficiente para desalojarlo del mínimo global

    - Empieza sacudiendo con fuerza, y luego gradualmente reduce la intensidad de la sacudida

Búsqueda por haz local

- Guarda la pista de k-estados

- Comienza con estados generados aleatoriamente

- En cada paso, se generan todos los sucesores de los k estados

- Si alguno es un objetivo, paramos el algoritmo.

- La información útil es pasada entre los k hilos paralelos de búsqueda

Algoritmos genéticos

- Es una variante de la búsqueda de haz estocástica (escoge a k sucesores aleatorios)
- Los estados sucesores se generan combinando dos estados padres
- Comienzan con un conjunto de k estados generados aleatoriamente llamado **población**
- Cada **individuo** está representado como una cadena sobre un alfabeto finito
- Cada estado se tasa con la función de evaluación o **función idoneidad**
- Finalmente cada posición está sujeta a una **mutación** aleatoria con una pequeña probabilidad independientes

Palabras clave

Búsqueda local	funcionan con un solo estado actual y generalmente se mueve solo a los vecinos del estado
Ascensión de colinas de reinicio aleatorio	Si al principio no tiene éxito, intente otra vez
gradiente descendente	Minimizar el coste
Población	Conjunto de k estados generados aleatoriamente

4 - Búsqueda informada y exploración
4.5 – Agentes de búsqueda online y ambientes desconocidos

- Los **agentes de búsqueda offline** calculan la solución completa antes de poner un pie en el mundo real y luego ejecutan la solución sin recurrir a sus percepciones

- Un agente de **búsqueda online** funciona **intercambiando** cálculo y la acción: primero toma una acción, luego observa el entorno y calcula la siguiente acción

    - Es buena idea para dominios dinámicos o semidinámicos. Es una idea mejor para dominios estocásticos.

- La búsqueda online es una buena idea en dominios dinámicos y semidinámicos (donde hay una penalización por holgazanear y utilizar demasiado tiempo para calcular)

- La búsqueda online es aun mejor para dominios estocásticos

- Una búsqueda offline debería presentar un plan de contingencia exponencialmente grande que considere todos los acontecimientos posibles

- Una búsqueda online necesita solo considerar lo que realmente pasa

- La búsqueda online es una idea necesaria para un **problema de exploración**

    - En un problema de este tipo los estados y acciones son desconocidos por el agente; un agente en este estado debe usar sus acciones como experimentos para determinar qué hacer después y a partir de ahí debe intercalar cálculo y la acción

Problemas de búsqueda online

- Un problema de búsqueda online puede resolverse solamente por un agente que ejecute acciones, más que por un proceso puramente computacional

- Asumimos que el agente sabe lo siguiente:

    - ACCIONES (s), devuelve una lista de acciones permitidas en el estado s

    - Funciones de costo individual (c, a, s') //no puede usarse hasta que el agente sepa que s' es el resultado

    - TEST-OBJETIVO(s)

- El agente no puede tener acceso a los sucesores de un estado excepto si intenta realmente todas las acciones en ese estado

- Asumiremos que el agente puede reconocer siempre un estado que ha visitado anteriormente y que las acciones son deterministas

- El agente podría tener acceso a una función heurística admisible h(s) que estime la distancia del estado actual a un estado objetivo

- El costo es el costo total del camino por el que el agente viaja realmente, se compara con el costo del camino que el agente seguiría si supiera el espacio de búsqueda de antemano. Esta comparación se hace con **proporción competitiva** y queremos que sea tan pequeña como es posible

- Ningún algoritmo puede evitar callejones sin salida en todos los espacios de estados

- El **argumento del adversario** es que podemos imaginar un adversario que construye el espacio de estados, mientras el agente lo explora

    - Puede poner el objetivo y callejos sin salida donde le guste

- Los callejones sin salida son una dificultad para la exploración de un robot, para avanzar asumiremos que el espacio de estados es **seguramente explorable**

    - Algún estado objetivo es alcanzable desde cualquier estado alcanzable

- Incluso en entornos seguramente explorables no se puede garantizar ninguna proporción competitiva acotada si hay caminos de costo ilimitado

Agentes de búsqueda online

- Después de cada acción, un agente online recibe una percepción al decirle que estado ha alcanzado, de esta información puede aumentar su mapa del entorno

- El mapa actual se usa para decidir dónde ir después

- Un algoritmo online puede expandir solo el nodo que ocupa físicamente

- Para evitar viajar a través de todo el árbol para expandir el siguiente nodo, parece mejor expandir los nodos en un orden local, esta propiedad la tiene la búsqueda primero por profundidad

Búsqueda local online

- La **búsqueda de ascensión de colinas** tiene la propiedad de localidad en sus expansiones de nodos

    - No es muy útil en forma simple porque deja que el agente se sitúe en máximos locales con ningún movimiento que hacer

    - Los reinicios aleatorios no puede utilizarse, porque el agente no puede moverse a un nuevo estado

    - En vez de reinicios, podemos considerar el uso de un **camino aleatorio**

        - El camino aleatorio selecciona simplemente al azar una de las acciones disponibles del estado actual, se puede dar preferencia a las acciones que todavía no se han intentado

- Aumentar la ascensión de colinas con memoria resulta una aproximación más eficaz

    - La idea básica es almacenar la mejor estimación actual H(s) del costo para alcanzar el objetivo desde cada estado que ha visitado

    - El costo estimado para alcanzar el objetivo a través de un vecino s' es el consto para s' más cl costo estimado para conseguir un objetivo desde ahí

- El **esquema AA*TR** aprendiendo A* en tiempo real
    - Construye un mapa del entorno usando la tabla resultado

    - Actualiza el costo estimado para el estado que acaba de dejar y entonces escoge el movimiento "aparentemente mejor" según sus costos estimados actuales

    - Las acciones que todavía no se han intentado en un estado s siempre se supone que dirigen inmediatamente al objetivo con el costo menor posible, esto es **optimismo bajo incertidumbre**

        - Esto anima al agente a explorar nuevos y posibles caminos prometedores

    - Un agente AA*TR garantiza encontrar un objetivo en un entorno seguramente explorable y finito, no es completo para espacios de estados infinitos

Aprendizaje en la búsqueda online

- La ignorancia de los agentes de búsqueda online proporcionan varias oportunidades de aprender

- Primero, los agente aprenden un "mapa" del entorno (resultado de cada acción en cada estado) registrando cada una de sus experiencias

- Segundo, los agentes de búsqueda locales adquieren estimaciones más exactas del valor de cada estado usando las reglas de actualización local, como AA*TR

    - Las actualizaciones convergen finalmente a valores exactos para cada estado

    - Una vez que se conoces los valores exactos, se pueden tomar decisiones óptimas simplemente moviéndose al sucesor con el valor más alto

Palabras clave

Búsqueda offline	calcular la solución completa antes de poner un pie en el mundo real y luego ejecutan la solución sin recurrir a sus percepciones
Búsqueda online	Intercambiar cálculo y acción
Problemas de exploración	un problema de este tipo los estados y acciones son desconocidos por el agente
Argumento de adversario	es que podemos imaginar un adversario que construye el espacio de estados, mientras el agente lo explora Puede poner callejones sin salida donde guste
seguramente explorable	Algún estado objetivo es alcanzable desde cualquier estado alcanzable
Camino aleatorio	Selecciona simplemente al azar una de

	las acciones disponibles del estado actual, se puede dar preferencia a las acciones que todavía no se han intentado
AA*TR	Construye un mapa del entorno usando la tabla resultado
Optimismo bajo la incertidumbre	Las acciones que todavía no se han intentado en un estado s siempre se supone que dirigen inmediatamente al objetivo con el costo menor posible

Capítulo 5 – problemas de satisfacción de restricciones
5.1 – Problemas de satisfacción de restricciones

- Tratar estados como más que sólo pequeñas cajas negras conduce a la invención de una nueva variedad de métodos de búsqueda

- Los capítulos anteriores exploraron la idea de que los problemas puedes resolverse buscando en un espacio de estados.

    - Los estados pueden evaluarse con heurísticas específicas del dominio y probados para ver si son estados objetivo

    - Desde el punto de vista del algoritmo de búsqueda, cada estado es una **caja negra**

        - Solo se le puede acceder con las rutinas especificas del problema

- En este capítulo se examinan **problemas de satisfacción de restricciones**
    - Los estados y test objetivo forman una representación simple, estándar y estructurada

    - Se utilizan heurísticas de propósito general mas que heurísticas especificas al problema

Problemas de satisfacción de restricciones

- Un PSR está definido por
    - un conjunto de variables X1, X2…Xn

    - Un conjunto de restricciones C1, C2..Cm

- Cada variable X tiene un **dominio** de **valores** posibles no vacío

- Cada restricción implica algún subconjunto de variables y especifica las combinaciones aceptables de valores para ese subconjunto

- Un estado del problema está definido por una **asignación** de valores a una o todas las variables

    - Una asignación que no viola ninguna restricción es asignación **consistente**

    - Una asignación completa es una asignación en la que se menciona cada variable

    - Una **solución de un PSR** es una asignación completa que satisface todas las restricciones

- Es bueno visualizar un PSR como un **grafo de restricciones**
    - Los nodos del grafo corresponden a variables del problema y los arcos corresponden a restricciones

- Al PSR se le puede dar una **formulación incremental** como en un problema de búsqueda estándar:
    - Estado inicial: asignación vacía { }, todas las variables no están asignadas

    - Función de sucesor: un valor se puede asignar a cualquier variable no asignada

    - Test objetivo: asignación actual es completa

    - Costo del camino: un costo constante para cada paso

- Cada solución debe ser una asignación completo, aparecen a profundidad n si hay n variables

    - Los algoritmos de búsqueda primero en profundidad son populares para PSRs

- El camino que alcanza una solución es irrelevante, podemos usar una **formulación completa de estados**

    - Cada estado es un asignación completa que podría o no satisfacer las restricciones

- La clase más simple de PSR implica variables discretas dominios finitos

- El número de posibles asignaciones completas es exponencial en el número de variables

- Los PSR con dominio finito incluyen **PSRs booleanos**

    - Las variables pueden ser verdaderas o falsas

- Las variables discretas también pueden tener **dominios infinitos**

    - No se puede describir restricciones enumerando todas las combinaciones permitidas de valores

    - En vez de eso, se usa un **lenguaje de restricción**

- Los problemas de satisfacción de restricciones con dominios continuos son muy comunes en el mundo real y son ampliamente estudiados en IO

    - Los **problemas de programación lineal** son la categoría más conocida de PSRs en dominios continuos
        - Las restricciones deben ser desigualdades lineales

- Además de tipos de variables, hay tipos de restricciones
    - **Restricción unaria:** restringe los valores de una sola variable

    - **Restricción binaria:** relaciona dos variables

Palabras clave

Caja negra	Estado sin estructura perceptible interna. Estructura de datos arbitraria a la que se puede acceder sólo con las rutinas especificas del problema
Problemas de satisfacción de restricciones	Definido por un conjunto de variables y restricciones
Consistente	Asignación que no viola ninguna restricción
Grafo de restricciones	Los nodos del grafo corresponden a variables del problema y los arcos corresponden a restricciones
PSRs booleanos	PSR donde las variables pueden ser verdaderas o falsas
Restricción unaria	restringe los valores de una sola variable
Restricción binaria	Relaciona dos variables

Capítulo 5 – problemas de satisfacción de restricciones
5.2 – Búsqueda con vuelta atrás para PSR

- Un problema es conmutativo si el orden de aplicación de cualquier conjunto de acciones no tiene ningún efecto sobre el resultado

- Todos los algoritmos de búsqueda para el PSR generan los sucesores considerando asignaciones posibles para sólo una variable en cada nodo del árbol de búsqueda

- La **búsqueda con vuelta atrás** elige valores para una variable a la vez y vuelve atrás cuando una variable no tiene ningún valor legal para asignarle

Variable y ordenamiento de valor

- La heurística de **mínimos valores restantes** escoge una variable que con mayor probabilidad causará pronto un fracaso

    - Si hay una variable X con cero valores legales restantes, la MVR seleccionará X y el fallo será descubierto inmediatamente

- El **grado heurístico** intenta reducir el factor de ramificación sobre futuras opciones seleccionando la variable que esté implicada en el mayor número de restricciones

- Una vez seleccionada la variable, el algoritmo debe decidir el orden para examinar sus valores, para esto se utiliza la heurística del **valor menos restringido**

    - Se prefiere el valor que excluye las pocas opciones de las variables vecinas del grafo de restricciones

Propagación de la información a través de las restricciones

- Hasta ahora el algoritmo de búsqueda considera las restricciones sobre una variable solo cuando la variable es elegida por SELECCIONA-VARIABLE-NOASIGNADA

- Mirando algunas restricciones antes en la búsqueda, podemos reducir drásticamente el espacio de ésta

Comprobación hacia delante

- Otra manera para usar mejor las restricciones durante la búsqueda se llama **comprobación hacia delante**

  - Siempre que se asigna una variable X, el proceso de comprobación hacia delante mira cada variable no asignada Y
    - y que esté relacionada con X por una restricción
    - y suprime el dominio de Y
    - Cualquier valor que sea inconsistente con el elegido para X también se suprime

Propagación de restricciones

- La **propagación de restricciones** es el término general para la propagación de las implicaciones de una restricción sobre una variable en las otras variables

- Un **arco consistente** proporciona un método rápido de propagación de restricciones

- Las formas más potentes de la propagación pueden definirse usando noción llamada **k-consistencia**

  - Un PSR es k-consistente si, para cualquier conjunto de k-1 variables, y ara cualquier asignación consistente a esas variables, siempre se puede asignar un valor consistente a cualquier k-ésima variable

- Un grafo es **fuertemente k-consistente** si es k-consistente y también (k-1) consistente, (k-2)… hasta 1-consistente

- Para cada variable Xi, tenemos solo que averiguar los valores de d, en el dominio, para encontrar un valor consistente

Vuelta atrás inteligente: mirando hacia atrás

- Cuando falla una rama de búsqueda, se va a atrás hasta la variable anterior e intenta un valor diferente para ella, **vuelta atrás cronológicamente**

- Una aproximación más inteligente a la vuelta atrás es ir hacia atrás hasta el **conjunto conflicto,** variables que causaron el fracaso

    - Para una variable X, es el conjunto de variables previamente asignadas que están relacionadas con X por las restricciones

    - El **método salto-atrás** retrocede a la variable más reciente en el conjunto conflictivo

- **Cómo se calcula el conjunto conflicto?**
    - El fracaso terminal de una rama de búsqueda siempre ocurre porque el dominio de una variable se hace vacío

Palabras clave

conmutatividad	el orden de aplicación de cualquier conjunto de acciones no tiene ningún efecto sobre el resultado
búsqueda con vuelta atrás	elige valores para una variable a la vez y vuelve atrás cuando una variable no tiene ningún valor legal para asignarle
Mínimos valores restantes	escoge una variable que con mayor probabilidad causará pronto un fracaso
Valor menos restringido	Se prefiere el valor que excluye las pocas opciones de las variables vecinas del grafo de restricciones
Comprobación hacia adelante	Manera mejor de usar las restricciones mientras se realiza la búsqueda
Propagación de restricciones	Es el término general para la propagación implicaciones de una restricción sobre una variable en las otras variables
Arco consistente	Proporciona un método rápido de propagación de restricciones
Vuelta atrás cronológicamente	Cuando falla una rama de búsqueda, se va a atrás hasta la variable anterior e intenta un valor diferente para ella Se visita el punto de decisión más reciente
conjunto conflicto	Variables que causaron el fracaso

Capítulo 5 – problemas de satisfacción de restricciones
5.3 – Búsqueda local para problemas de satisfacción de restricciones

- Los algoritmos de búsqueda local resultan ser muy eficientes en la resolución de PSRs

    - Usan una formulación estados completa: el estado inicial asigna un valor a cada variable

    - La función sucesor trabaja cambiando el valor de una variable a la vez

- En la elección de un nuevo valor para una variable, la heurística debe seleccionar el valor que cause el número mínimo de conflictos con otras variables (heurística de **mínimos-conflictos**)

- Otra ventaja de búsqueda local es que puede usarse en un ajuste online cuando el problema cambia

Capítulo 5 – problemas de satisfacción de restricciones
5.4 – La estructura de los problemas

- **Subproblemas independientes:** cualquier solución para una variable y para la otra variable produce una solución para el mapa entero)

- Cualquier PSR estructurado por un árbol puede resolverse en tiempo lineal en el número de variables

- **Descomposición en árbol** es resolver cada subproblema independientemente, y las soluciones que resultan son entonces combinadas

    - Satisface las tres exigencias siguientes:
        - Cada variable en el problema original aparece en al menos uno de los subproblemas

        - Si dos variables están relacionadas por una restricción del problema original, deben aparecer juntas en al menos uno de los subproblemas

        - Si una variable aparece en dos subproblemas en el árbol, debe aparecer en cada subproblema a lo largo del camino que une a esos subproblemas

            - Cualquier variable debe tener el mismo valor en cada subproblema en el cual aparece

- Si se puede resolver cada subproblema independientemente entonces podemos construir una solución global, de lo contrario no tiene solución

    - Vemos cada subproblema como una “megavariable” cuyo dominio es el conjunto de todas las soluciones para el problema