**Introducción a los Sistemas Basados en el Conocimiento**

Resolución de problemas y conocimiento

Los métodos de resolución de problemas que hemos visto son de aplicación general, se fundamentan en una función heurística para obtener el orden de exploración de soluciones. La capacidad expresiva de los heurísticos es reducida, una única función no puede representar todas las decisiones de exploración en el problema. El ahorro en coste computacional es limitado y con conocimiento más específico se podrían tomar mejores decisiones.

De los sistemas Expertos a los SBC

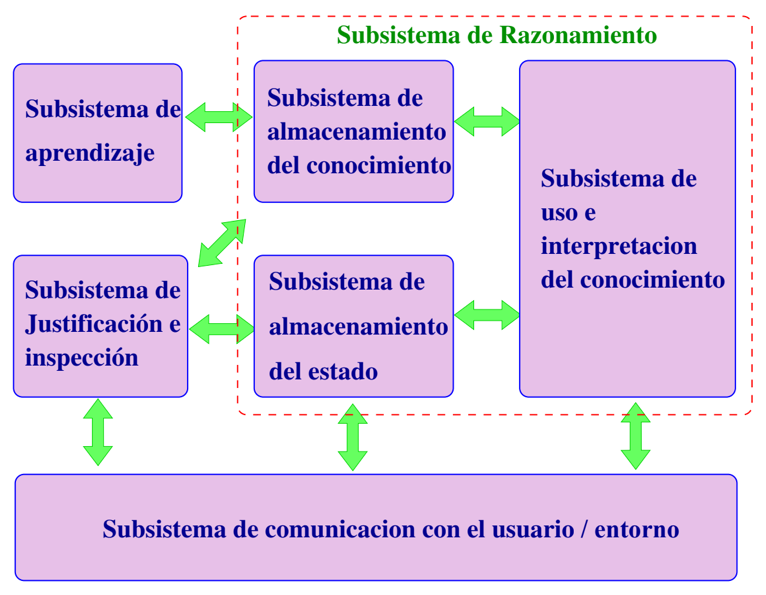
Sistemas Expertos:

* El objetivo es emular la capacidad de resolución de expertos humanos.
* Se construyen por procesos de ingeniería del conocimiento.
* Basados principalmente en sistemas de reglas de producción.
* Sistemas cerrados con poca capacidad de aprendizaje.

Sistemas Basados en el Conocimiento:

* El objetivo es usar conocimiento del dominio para solucionar problemas.
* Incluyen procesos automáticos de adquisición del conocimiento al proceso de ingeniería del conocimiento.
* Metodologías y arquitecturas heterogéneas (reglas, casos, modelos cualitativos, agentes inteligentes, computación emergente, …).
* Sistemas adaptables con capacidad de aprendizaje.

Componentes de los SBC



Características de los SBC

Los SBC se aplican a problemas complejos donde los sistemas convencionales de software no son suficientes.

La naturaleza de estos problemas requiere ciertas características:

* Flexibilidad para abordar diferentes problemas.
* Emulación de comportamiento racional como mecanismo de resolución.
* Operar en un entorno rico y con mucha información.
* Uso de información simbólica en el proceso de razonamiento.
* Uso de interfaces naturales en su comunicación con el usuario.
* Capacidad de aprendizaje como método de adaptación.

En la construcción de sistemas con estas características:

* Deberemos mantener el conocimiento del dominio y de resolución de problemas separado del mecanismo de control de la resolución.
* Deberemos incorporar conocimiento heurístico en la resolución (incompleto, aproximado, no sistemático).
* Deberemos permitir una interacción estrecha con el usuario y/o entorno.

Área de la IA involucradas en los SBC

Las características que buscamos en los SBC y el tipo de habilidades que necesitan, hacen que se combinen diferentes áreas de la inteligencia artificial en su diseño y construcción, entre ellas:

* Representación del conocimiento (conocimiento sobre dominio, sobre la resolución, heurísticas, …).
* Razonamiento e inferencia (lógica clásica, incompletitud, incertidumbre, tiempo, …).
* Búsqueda heurística / Resolución de problemas.
* Tratamiento del lenguaje natural (interfaces).
* Aprendizaje automático (adquisición del conocimiento del dominio, adaptación, …).

Necesidades de los SBC

* Disponer del conocimiento de expertos altamente cualificados.
* Poder ayudar / formar expertos / no expertos.
* Preservar el conocimiento de expertos.
* Obtener soluciones rápidas y justificadas.
* Tratar grandes volúmenes de información.
* Tener sistemas que tomen decisiones autónomas.

Problemas resolubles mediante SBC

* Han de tener una complejidad suficiente que justifique el coste de su desarrollo.
* Han de estar correctamente dimensionados.
* Hemos de poder disponer del conocimiento experto necesario.
* Hemos de poder plantear el problema como un proceso de razonamiento.
* Han de poder estructurarse adecuadamente.
* No tiene una solución mediante técnicas tradicionales.
* Tenemos expertos cooperativos.

Problemas de los SBC

* Fragilidad.
* Dificultad del control del razonamiento.
* Poca reusabilidad del conocimiento o las estrategias de resolución.
* Difícil integración del aprendizaje en el sistema.
* Dificultad de adquirir el conocimiento a partir de expertos.
* Dificultad en la validación de la correctitud / completitud del sistema.

Áreas de aplicación de los SBC

* Se pueden encontrar en cualquier dominio en el que se necesite un conocimiento especializado.
* Existen aplicaciones en multitud de dominios (medicina, ingeniería, predicción meteorológica, banca, …).
* Problemas que involucran el análisis de un conjunto de evidencias (interpretación, diagnóstico, supervisión, predicción, …).
* Problemas que involucran la construcción de una solución (diseño, planificación, configuración, …).

**Representación del conocimiento**

Todo problema es más sencillo de resolver si disponemos de conocimiento específico sobre él. Este conocimiento dependiente del dominio se combina con el conocimiento general sobre cómo resolver problemas i ha de permitir guiar a los mecanismos de IA para obtener soluciones de manera más eficiente.

Problemas:

* ¿Cómo escoger el formalismo de representación que nos permita hacer una traducción fácil del mundo real a la representación?
* ¿Cómo ha de ser esa representación para que pueda ser utilizada de forma eficiente?

Información -> conjunto de datos básicos, sin interpretar, que se obtiene como entrada del sistema. Por ejemplo:

* Los datos numéricos que aparecen en una analítica de sangre.
* Los datos de los sensores de una planta química.

Conocimiento -> conjunto de datos de primer orden, que modelan de forma estructurada la experiencia que se tiene sobre un cierto dominio o que surgen de interpretar los datos básicos. Por ejemplo:

* La interpretación de los valores de la analítica de sangre o de los sensores de la planta química para decir si son normales, altos o bajos, preocupantes, peligrosos, …
* El conjunto de estructuras de datos y métodos para diagnosticar a pacientes a partir de la interpretación del análisis de sangre, o para ayudar en la toma de decisiones de que hacer en la planta química.

Los sistemas de IA necesitan diferentes tipos de conocimiento que no suelen estar disponibles en bases de datos y otras fuentes de información:

* Conocimiento sobre los objetos en un entorno y posibles relaciones entre ellos.
* Conocimiento sobre los procesos en los que interviene o que le son útiles.
* Conocimiento difícil de representar como datos básicos, como la intencionalidad, la causalidad, los objetos, información temporal, conocimiento que para los humanos es de “sentido común”, etc.

Intuitivamente se puede decir:

Conocimiento = Información + Interpretación

Para representar algo necesitamos saber:

* Su forma o estructura.
* Que uso le dan los seres inteligentes.
* Que uso le dará una inteligencia artificial.
* Como adquirir el conocimiento.
* Como almacenarlo y manipularlo.

Por desgracia no hay respuestas completas para todas estas preguntas desde le punto de vista biológico o neurofisiológico. Se intenta construir modelos que simulen la adquisición, estructuración y manipulación del conocimiento y que nos permitan crear sistemas artificiales inteligentes.

Esquema de representación

Es un instrumento para codificar la realidad en un ordenador. Es importante distinguir entre:

* Mundo real (lo que queremos representar) -> Dominio.
* Su representación -> uno o más esquemas de representación.

Desde un punto de vista informático, un esquema de representación puede ser descrito como una combinación de:

* Estructuras de datos que codifican el problema en curso con el que se enfrenta el agente -> parte estática.
* Estructuras de datos que almacenan conocimiento referente al entorno en el que se desarrolla el problema y procedimientos que manipulan las estructuras de forma consistente con una interpretación plausible de las mismas -> parte dinámica.

La parte estática está formada por:

* Estructura de datos que codifica el problema.
* Operaciones que permiten crear, modificar y destruir elementos en la estructura.
* Predicados que dan un mecanismo para consultar esta estructura de datos.
* Semántica de la estructura: se necesita definir la relación entre la realidad y la representación escogida.

R(Elemento estructura, Mundo Real)

La parte dinámica está formada por:

* Estructuras de datos que almacenan conocimiento referente al entorno / dominio en el que se desarrolla el problema.
* Procedimientos que permiten:
  + Interpretar los datos del problema (de la parte estática) a partir del conocimiento del dominio (de la parte dinámica).
  + Controlar el uso de los datos: estrategias de control.
  + Adquirir nuevo conocimiento.

Se ha de tener siempre en cuenta que nuestra representación siempre es incompleta, debido a:

* Modificaciones: el mundo es cambiante, pero nuestras representaciones son de un instante.
* Volumen: mucho (demasiado) conocimiento a representar -> representación parcial.
* Complejidad: la realidad tiene una gran riqueza en detalles.

El problema de modificación del mundo está ligado a los procedimientos de adquisición y mantenimiento de la representación (Frame Problem).

Los problemas de volumen y complejidad de la realidad están relacionados con la granularidad de la representación.

Propiedades de un sistema de representación

Un sistema de representación debe poseer las siguientes propiedades:

* Ligados a la representación:
  + Adecuación Representacional: habilidad para representar todas las clases de conocimiento que son necesarias en aquel dominio.
  + Adecuación Inferencial: habilidad de manipular estructuras de representación de tal manera que devengan o generen nuevas estructuras que correspondan a nuevos conocimientos inferidos de los anteriores.
* Ligados al uso de la representación:
  + Eficiencia Inferencial: capacidad del sistema para incorporar información adicional a la estructura de representación, llamada metaconocimiento, que puede emplearse para focalizar la atención de los mecanismos de inferencia con el fin de optimizar los cómputos.
  + Eficiencia en la Adquisición: capacidad de incorporar fácilmente nueva información. Idealmente el sistema por sí mismo deberá ser capaz de controlar la adquisición de nueva información y su posterior representación.

Tipos de conocimiento

1. Conocimiento Declarativo:
   * El conocimiento se representa de forma independiente a su uso posterior.
   * El control del uso adecuado se logra:
     + Mediante heurísticas de propósito general que determina la mejor manera de usar el conocimiento.
     + Mediante la adición de información sobre el control del uso del conocimiento declarativo que dirija al mecanismo de resolución.
   * Tipos de conocimiento declarativo:
     + Conocimiento relacional.
     + Conocimiento heredable.
     + Conocimiento inferible.
2. Conocimiento Procedimental:
   * El conocimiento representado implica la inclusión de información sobre cómo usarlo.

*Conocimiento Relacional Simple:*

La forma más simple de representar hechos declarativos es mediante un conjunto de relaciones expresables mediante tablas (como en una Base de Datos).

Problema: tal cual no aporta mucha información.

Hemos de aportar procedimientos que lo enriquezcan -> Motor de inferencia: genera conocimiento a partir de información.

Las Bases de Datos pueden proporcionar información a los SBC.

*Conocimiento Heredable:*

Suele ser muy útil el disponer de una estructuración jerárquica del conocimiento (taxonomía jerárquica). Se trata de construir un árbol o grafo de conceptos basado en la generalización y/o especialización:

* Los nodos son los conceptos/clases. Los arcos son las relaciones:
  + is-a (es un): relación clase-clase.
  + Instance-of (instancia de, ejemplar de): relación clase-ejemplar.

El mecanismo de inferencia es la herencia de propiedades y valores:

* Herencia simple/múltiple.
* Valores por defecto.

*Conocimiento Inferible*

Conocimiento descrito mediante lógica. Se puede utilizar la semántica de los operadores y el Modus Ponens para inferir nuevo conocimiento. El mecanismo de inferencia en el caso de la lógica de primer orden se obtiene eligiendo entre los métodos generales de resolución automática de teoremas que existen.

*Conocimiento Procedimental*

Conocimiento que, a diferencia del declarativo, incluye la especificación de los procesos de uso del conocimiento:

* Programas: utilizan funciones para obtener el conocimiento a partir de información de otro conocimiento que ya se tiene.
* Reglas de producción: si se cumplen unas condiciones entonces se realizan unas acciones u otras.

Este tipo de conocimiento suele ser más eficiente computacionalmente, pero hace más difícil la inferencia y la adquisición/modificación.

**Sistemas de producción**

Introducción

Las representaciones mediante formalismos lógicos pueden verse de forma procedimental, describimos cuales son los pasos para resolver un problema como una cadena de deducciones.

La representación se basa en dos elementos:

* Hechos: proposiciones o predicados.
* Reglas: fórmulas condicionales donde el consecuente habitualmente está formado por un predicado atómico o una acción.

Analogía con búsqueda en espacio de estados:

* Hechos = estado del problema.
* Reglas = operadores de búsqueda.

Un problema quedará definido por:

* Base de Hechos: predicados que describen el problema concreto.
* Base de conocimiento (o de reglas): reglas que describen los mecanismos de razonamiento que permiten resolver problemas.
* Motor de inferencia: ejecuta el formalismo y obtiene la cadena de razonamiento que soluciona el problema.

Motor de inferencias

Funciones:

* Deducir nuevos hechos, ejecutar acciones para resolver el problema (objetivo) a partir de un conjunto inicial de hechos y eventualmente a través de una interacción con el usuario.

Componentes:

* Intérprete de reglas + estrategia de control.

Fases:

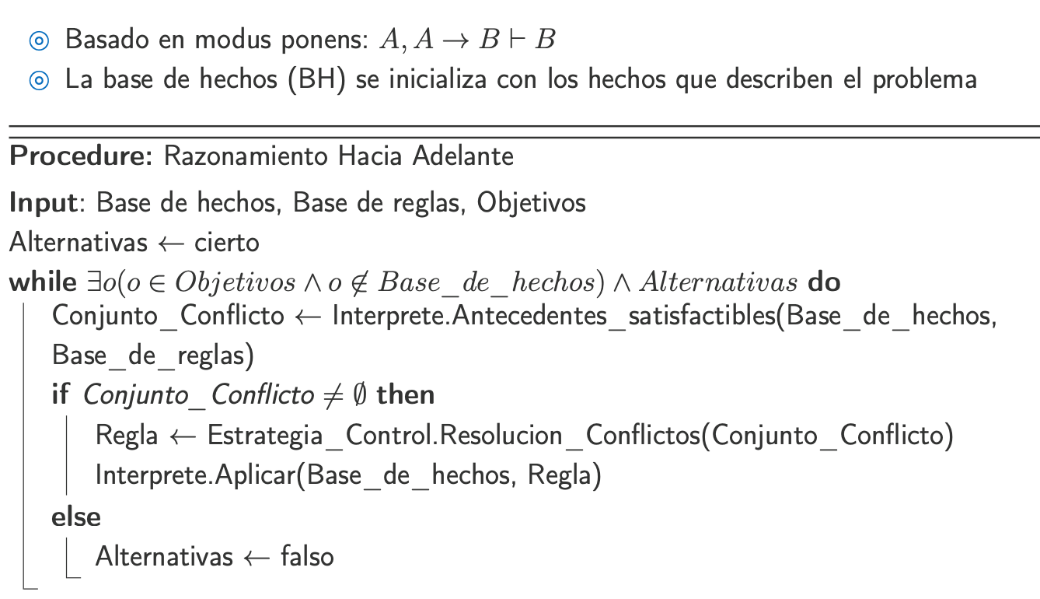
1. Detección (filtro): REGLAS PERTINENTES
   * Obtención / construcción del conjunto de instanciaciones de reglas. Formación del conjunto de conflictos.
   * El intérprete de reglas realiza los cálculos e instanciaciones necesarias que son posibles en cada estado de resolución del problema (matching).
   * Una regla puede instanciarse más de una vez, caso de existir variables (p. ej. CP1).
2. Selección: ¿QUÉ REGLA?
   * Resolución de conflictos: selección de la instanciación (la mejor) a aplicar.
   * Las reglas instanciadas son seleccionadas para aplicarse dependiendo de la estrategia de control (Estrategia de resolución de conflictos):
     + Estrategia fija.
     + Estrategia dinámica prefijada.
     + Estrategia guiada por meta-reglas.
   * Criterios aplicables:
     + 1ª regla por orden en la Base de Conocimientos.
     + La regla más / menos utilizada.
     + La regla más específica / la más general.
     + La regla que tenga el grado de certeza más alto.
     + La instanciación que satisfaga los hechos (más prioritarios, más antiguos, más nuevos…).
3. Aplicación:
   * Aplicación / ejecución de la regla:
     + Modificación de la base de hechos.
     + Nuevos cálculos, nuevas acciones, preguntas al usuario.
     + Nuevos subojetivos.
   * Propagación de las instanciaciones (en CP1).
   * Propagación del grado de certeza.
   * El proceso de deducción acaba cuando:
     + Se encuentra la conclusión (el objetivo) buscado -> éxito.
     + No queda ninguna regla aplicable -> ¿fracaso?

Razonamiento

Hay diferentes tipos de razonamiento:

* Deductivos, progresivos, forward chaining (FC), encadenamiento dirigido por hechos.
  + Evidencia, síntomas, datos -> conclusiones, hipótesis.
* Inductivos, regresivos, backward chaining (BC), encadenamiento dirigido por objetivos.
  + Conclusiones, hipótesis -> datos, evidencias, síntomas.
* Mixtos, encadenamiento híbrido.

*Encadenamiento dirigido por hechos*



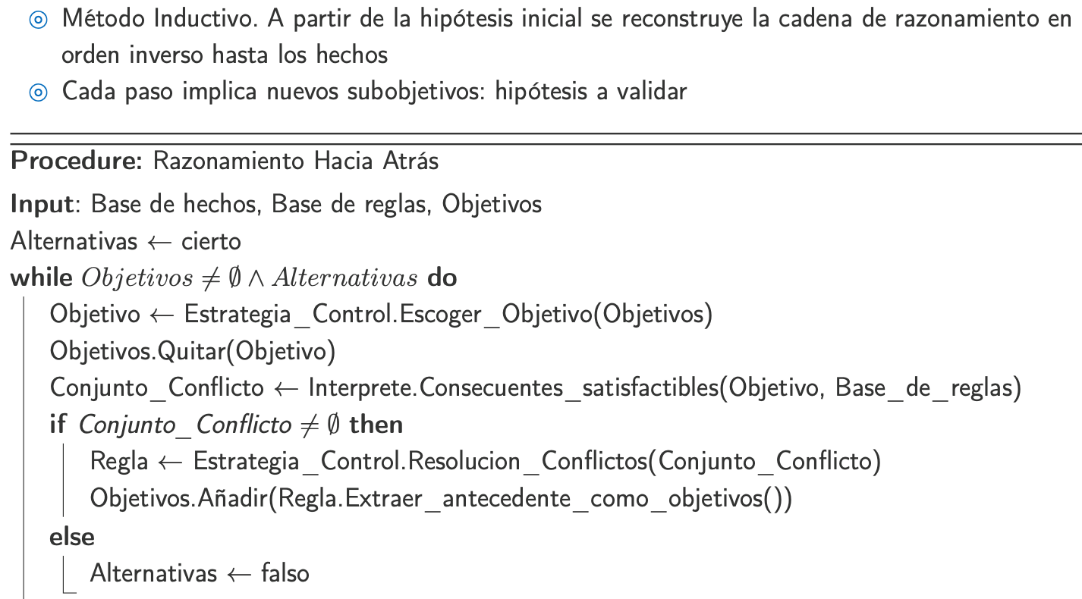
Problemas:

* La búsqueda no está localizada en el objetivo.
* Explosión combinatoria, deducimos hechos no relacionados con la solución.

Ventajas:

* Deducción intuitiva.
* Facilita la formalización del conocimiento al hacer un uso natural del mismo.
* Puede ser usado de manera exploratoria.

*Encadenamiento dirigido por objetivos*



La resolución se plantea como una descomposición en subproblemas. El proceso de resolución consiste en la exploración de un árbol Y / O.

Ventajas:

* Sólo se considera lo necesario para la resolución del problema.

Desventajas:

* Hemos de conocer la solución del problema a priori.

*Encadenamiento híbrido*

Partes de la cadena de razonamiento que conduce de los hechos a los objetivos se construyen deductivamente y otras inductivamente. Es decir, hace una exploración bidireccional. El cambio de estrategias suele llevarse a cabo a través de meta-reglas. Se evita la explosión combinatoria del razonamiento deductivo y mejora la eficiencia del razonamiento inductivo cuando no existen objetivos claros,

**Ontologías**

El objeto de estudio de la ciencia de la ontología es el estudio de las categorías que existen en un dominio. El resultado de este estudio es lo que denominamos una ontología.

Una ontología es un catálogo de los tipos de cosas que asumimos que existen en un dominio D desde la perspectiva de alguien que usa un lenguaje L con el propósito de hablar de D.

Los elementos de una ontología representan predicados, constantes, conceptos y relaciones pertenecientes a un lenguaje L cuando se usa para comunicar información sobre D. Una ontología es pues un vocabulario.

La capacidad de obtener deducciones a partir de la información que representa la ontología viene dada por el uso de una lógica.

La lógica por si misma no habla sobre nada, es neutra respecto al significado, es su combinación con una ontología lo que le da a un formalismo lógico la capacidad de expresar significados. Si P -> Q ^ P, entonces Q, este razonamiento no habla sobre nada en concreto salvo que asignemos significados a los átomos (P = llueve, Q = me mojo).

Motivación

¿Cuáles son los puntos interesantes que motivan el uso de las ontologías?

* Permiten compartir la interpretación de la estructura de la información entre personas / agentes. El establecer una ontología sobre un dominio permite que dos agentes puedan entenderse sin ambigüedad y sepan a que se refieren.
* Permiten reusar el conocimiento. Hacer una descripción de un dominio permite que esta pueda ser usada por otras aplicaciones que necesitan tratar con ese conocimiento.
* Hacen que nuestras suposiciones sobre el dominio se hagan explicitas. Facilita replantearse las suposiciones sobre el dominio y ayuda a que otros puedan entender su descripción.
* Separan el conocimiento del dominio del conocimiento operacional. Permite hacer independientes las técnicas y algoritmos para solucionar un problema del conocimiento concreto del problema.
* Permiten analizar el conocimiento del dominio. Una vez tenemos una especificación del conocimiento podemos analizarlo utilizando métodos formales (para comprobar si es correcto, completo, …).

Desarrollo de ontologías

En nuestra área de trabajo (IA), una ontología será una descripción formal explicita de los conceptos de un dominio (Clases). Estas clases se describirán a partir de propiedades que representarán las características, atributos y relaciones de las clases. Adicionalmente estas características tendrán restricciones (tipo, cardinalidad, …). Finalmente, tendremos instancias (elementos identificables) que constituirán los individuos concretos que representan la ontología.

Desarrollar una ontología requerirá:

* Definir las clases que forman el dominio.
* Organizar las clases en una jerarquía taxonómica.
* Definir las propiedades de cada clase e indicar las restricciones de sus valores.
* Asignar valores a las propiedades para crear instancias.

No existe una metodología establecida sobre como desarrollar ontologías, pero se debe tener en cuenta:

* No existe un modo correcto de modelar un dominio. La mejor solución dependerá de la aplicación / problema concreto.
* El desarrollo de una ontología es un proceso iterativo.
* Los elementos de la ontología deberían ser cercanos a los conceptos y relaciones que se usan para describir el domino (generalmente se corresponden a nombres y verbos que aparecen en frases que describen el dominio).

Fases de desarrollo de una ontología:

FASE 1: Determinar el dominio y la cobertura de la ontología

* ¿Qué dominio cubrirá la ontología?
* ¿Para qué usaremos la ontología?
* ¿A que tipos de pregunta ha de poder responder la ontología?
* ¿Quién usará y mantendrá la ontología?

FASE 2: Considerar la reutilización de ontologías existentes

Las ontologías se construyen para comunicar conocimiento en dominios, por lo que se construyen con la idea de compartición. No es necesario rehacer un trabajo que ya está hecho, si existe una ontología sobre el dominio en el que trabajamos, podemos incorporarla.

FASE 3: Enumerar los términos importantes en la ontología

Escribir una lista de términos que podemos usar para referirnos a nuestro dominio, elaborando frases que podríamos utilizar para preguntarnos cosas sobre él o para explicar a alguien información sobre él.

* ¿Qué propiedades tiene esos términos?
* ¿Qué nos gustaría decir sobre ellos?

FASE 4: Definir las clases y su jerarquía

Podemos tomar diferentes aproximaciones:

* De arriba a abajo: definimos los conceptos más generales y vamos especializándolos.
* De abajo a arriba: definimos las clases más específicas y vamos agrupándolas según propiedades comunes generalizando.
* Combinación de ambas: definimos los conceptos más importantes y especializamos y generalizamos para completar la ontología.

Ninguno de estos métodos es esencialmente mejor y depende del dominio.

FASE 5: Definir las propiedades de las clases

* Debemos describir la estructura interna de las clases.
* Determinaremos una lista de características y en que clases debemos tenerlas.
* Podemos tener muchos tipos de propiedades:
  + Propiedades descriptivas, cualidades.
  + Propiedades identificadoras, nombres.
  + Partes.
  + Relaciones con instancias de otras clases.
* Las propiedades deberían asignarse a la clase más general, el resto las obtendrán vía herencia.

FASE 6: Definir las características de las propiedades

* Cardinalidad (número de valores permitidos).
* Tipo, valores.
* Valores por defecto.
* Obligatoriedad.
* Si es una relación definir la cardinalidad y el rango.

FASE 7: Crear instancias

Si es necesario (en algún momento lo será) crear las instancias que formarán parte de la ontología.

Consejos para el desarrollo de una ontología:

* No incluir versiones singulares y plurales de un termino (la mejor política es usar solamente nombres en singular o plural).
* Los nombres no son las clases, debemos distinguir la clase del nombre que le damos. Podemos tener sinónimos, pero todos representan a la misma clase.
* Asegurarnos de que la jerarquía está correctamente construida.
* Observar las relaciones de transitividad y comprobar si son correctas.
* Evitar ciclos en la jerarquía.
* Todas las subclases de una clase deben estar al mismo nivel de generalidad.
* No hay un criterio respecto al número de clases, la experiencia dice que un número entre dos y doce es habitual, más clases indicaría que tenemos que estructurarlas añadiendo más niveles.
* ¿Cuándo introducir nuevas clases? Suele ser incomodo navegar por jerarquías o muy planas o muy profundas, se debería elegir un punto intermedio, unas indicaciones serian:
  + Las nuevas clases tienen propiedades adicionales que no tiene la superclase.
  + Tienen restricciones diferentes.
  + Participan en relaciones diferentes.
* Decidir si hemos de usar una propiedad o crear una clase. A veces un atributo es suficientemente importante como para considerar que sus valores diferentes corresponden a objetos diferentes.
* Decidir dones está el nivel de las instancias. Pensar cual es el nivel mínimo de granularidad que necesitamos.
* Limitar el ámbito de la ontología;
  + La ontología o necesita incluir todas las clases posibles del dominio, o solo las necesarias para la aplicación que se desarrolla.
  + Tampoco necesitamos incluir todos los atributos / restricciones / relaciones posibles.

**Representación del Conocimiento en SBCs**

Componentes de los SBC

Diagrama

Descripción generada automáticamente

SBC basados en sistemas de producción

La resolución se obtiene a partir del proceso de razonamiento de un motor de inferencia.

El conocimiento del dominio está expresado mediante una ontología y el conocimiento de resolución de problemas está almacenado habitualmente como reglas de producción o un formalismo equivalente.

*Almacenamiento del conocimiento*

Almacenará todo el conocimiento para resolver problemas en el dominio de aplicación. Encontraremos tres tipos de conocimiento:

* Conocimiento factual (objetos del dominio y sus características).
* Conocimiento relacional (relaciones entre los objetos del dominio).
* Conocimiento condicional (conocimiento deductivo sobre el problema).

Los dos primeros conocimientos están descritos mediante la ontología de dominio y el tercer conocimiento describirá el conocimiento relacionado con la resolución.

El conocimiento condicional incluye:

* Conocimiento deductivo (estructural): describe los procesos de resolución de problemas como cadenas de deducción.
* Conocimiento sobre objetivos (estratégico): orienta el proceso de resolución.
* Conocimiento causal (de soporte): apoya al proceso de explicación de la resolución.

*Módulos de reglas:*

* Permite facilitar el desarrollo y el mantenimiento del sistema.
* Permite aumentar la eficiencia del proceso de razonamiento.
* Permite implementar estrategias de uso del conocimiento (metaconocimiento, meta-reglas).

*Meta-reglas:*

Describen conocimiento a alto nivel sobre la resolución del problema. Permiten dirigir el control de la resolución:

* Activar y desactivar reglas/módulos.
* Decidir el orden de ejecución de reglas/módulos.
* Decidir estrategias de resolución. Tratamiento de excepciones, incertidumbre, …

Son más difíciles de obtener de los expertos.

*Uso e interpretación del conocimiento*

Es habitualmente un motor de inferencia y aplicará su ciclo de ejecución para resolver el problema:

* Detección de reglas aplicables.
* Selección de la mejor regla (estrategia general o guiada por el metaconocimiento).
* Aplicación de la regla.

*Almacenamiento del estado*

Guarda los datos iniciales del problema y los hechos obtenidos durante el proceso de resolución. Puede guardar otro tipo de información necesaria para el control de la resolución y otros subsistemas:

* Orden de deducción de los hechos.
* Preferencias sobre el uso de los hechos.
* Reglas que generan los hechos.
* Reglas activadas recientemente.
* Puntos de backtracking.
* …

*Justificación de la solución*

La posibilidad de justificar las decisiones da credibilidad al sistema. También permite detectar deducciones erróneas. Un sistema debería poder contestar Porqué y Cómo. Hay diferentes niveles de justificación:

* Muestra: Traza de los pasos de resolución.
* Justificación: Razones de los elementos que aparecen en la traza de resolución (línea de razonamiento, preguntas, hechos, preferencias, subproblemas, …).

*Aprendizaje*

Por lo general, el conjunto de problemas que se resuelven está acotado. En algunos dominios es necesario adaptarse al entorno y resolver nuevos problemas. El aprendizaje puede suceder:

* Durante el proceso de construcción del SBC: se substituye o complementa el proceso de adquisición con métodos de aprendizaje inductivo, se construye un modelo a partir de ejemplos.
* Durante el proceso de resolución: se detectan y corrigen las resoluciones erróneas, se aprenden reglas de control que mejoran la eficiencia del proceso de resolución.

SBC basados en Razonamiento basado en casos

La resolución de un problema se obtiene identificando una solución anterior similar. Ventajas:

* Reducen el problema de extracción del conocimiento.
* Facilitan el mantenimiento/corrección/extensión del sistema.
* Permite una resolución más eficiente.
* Permite explicaciones más cercanas a la experiencia del usuario.

*Ciclo de ejecución*

Consta de cuatro fases:

1. Recuperación: búsqueda de los casos almacenados más similares.
2. Reuso: obtenemos la solución del caso recuperado.
3. Revisión: evaluamos y adaptamos la solución recuperada.
4. Retención: comprobamos si es interesante guardar el caso.

Diagrama

Descripción generada automáticamente

*Almacenamiento del conocimiento*

El conocimiento estará formado por casos. Un caso es una estructura compleja (características, solución). Se almacenarán en la base de casos (estructura, indexación). Tendremos también conocimiento para:

* Evaluar la similaridad entre los casos.
* Combinar/adaptar las soluciones recuperadas.
* Evaluar las soluciones.

*Uso e interpretación del conocimiento*

Se basa en el ciclo de ejecución de razonamiento basado en casos:

* Búsqueda en la base de casos de los casos más similares.
* Recuperación de las soluciones de los casos.
* Combinación/adaptación de soluciones (procedimientos/razonamiento).

*Almacenamiento del estado*

Información del caso actual, cálculo de los casos más similares y razonamiento para la evaluación/combinación/adaptación de las soluciones.

*Justificación*

Es parte de la información de los casos y se complementará con el razonamiento sobre la combinación/adaptación de las soluciones.

*Aprendizaje*

Añadir nuevos casos (más sencillo que en los sistemas de reglas), la solución debe ser suficientemente diferente (evaluación) y podemos olvidar casos (poco usados, parecidos a otros).

Otras metodologías

*Sistemas basados en redes neuronales*

Dentro del área de la IA conexionista. El elemento base es la neurona (elemento de cómputo).

Neurona -> entradas, salidas, estado, funciones para la combinación de las entradas y el estado y función para generar la salida.

Las neuronas se organizan en redes con diferentes capas y la red asocia unas entradas (datos del problema) a unas salidas (solución del problema) y se debe entrenar (ejemplos resueltos) para que aprenda a resolver el problema (asociación).

Diagrama

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente

*Razonamiento basado en modelos*

Construimos un modelo del comportamiento del sistema, este modelo se basa en información cualitativa. Razonando sobre el modelo podemos predecir las consecuencias de nuestras acciones. Utilizamos razonamiento de sentido común en la resolución.

*Agentes inteligentes/Sistemas multiagente*

Nos alejamos de una visión monolítica de los sistemas inteligentes. Un agente inteligente resuelve una tarea “sencilla”. El problema global se resuelve en cooperación/ coordinación. Otras áreas involucradas (organización, cooperación, coordinación, negociación, división del trabajo, comunicación, razonamiento sobre otros, …).

Ventajas: sistemas flexibles, reconfiguración/reorganización para otras tareas (componentes) -> resolver más problemas.

Relacionado con computación Grid y servicios web.

**Sistema Basado en el conocimiento (SBC)**

Desarrollo de SBC

El punto más importante del desarrollo de SBC es la extracción del conocimiento. Requiere la interacción entre el Ingeniero del Conocimiento y el experto.

La metodología de ingeniería de software han de encajar este proceso entre sus fases y han de adaptarse a las características específicas de los SBC.

Diagrama

Descripción generada automáticamente Imagen que contiene exterior

Descripción generada automáticamente

Modelo en cascada Modelo en espiral

Diferencias de los SBC

Sistema software convencionales -> algoritmos conocidos y de uso común. Es posible estimar la naturaleza y cantidad del conocimiento.

SBC -> conocimiento incompleto, impreciso, heurístico. Es difícil estimar la naturaleza y cantidad de conocimiento.

Es complicado obtener un diseño adecuado en las fases iniciales. Decisiones iniciales erróneas pueden provocar el replanteamiento radical del diseño durante el desarrollo.

El ingeniero del conocimiento debe realizar un proceso de adquisición del conocimiento, es decir, entrevistas con los expertos.

* El IC debe aprender los elementos básicos del dominio.
* Encontrar un formalismo representación que pueda entender el experto.
* Los expertos prefieren casos al razonamiento a partir de definiciones generales.
* A los expertos les es difícil explicitar su conocimiento en detalle (paradoja del experto).

Solución: Diseño incremental y prototipado rápido.

Objetivo: Desarrollar un prototipo funcional que recoja las funcionalidades básicas del sistema.

El análisis y la especificación deben tener en cuenta el sistema completo.

El diseño e implementación se limita al prototipo inicial.

Este prototipo se completa incrementalmente.

Ventaja: Disponemos de un sistema funcional durante todo el proceso.

Ciclo de vida de un SBC

Texto

Descripción generada automáticamente

1. Análisis del problema: recopilar información sobre el proyecto y determinar su viabilidad.
2. Especificación de requerimientos: fijar los objetivos y métodos para conseguirlos.
3. Diseño preliminar: decisiones a alto nivel sobre el diseño (formalismo de representación del conocimiento, herramientas, fuentes de conocimiento).
4. Prototipo inicial y evaluación: construir un prototipo con cobertura limitada, evaluar las decisiones de diseño a partir del prototipo.
5. Diseño final: validar las decisiones y proponer el diseño del sistema de manera que permita un desarrollo incremental.
6. Implementación: completar la adquisición del conocimiento, ampliar incrementalmente el prototipo inicial.
7. Validación y verificación: comprobar que el sistema cumple las especificaciones.
8. Ajustes de diseño: realimentar el proceso (los cambios en el diseño deberían ser mínimos).
9. Mantenimiento: mantener el sistema.

Metodologías especializadas

CommonKADS:

* Ciclo de vida en espiral y modelado mediante herramientas parecidas a UML.
* Se construyen seis modelos: organización, tareas, agentes, comunicación, conocimiento y diseño.

MIKE:

* Ciclo de vida en espiral: adquisición del conocimiento (modelo de adquisición y modelo de estructura), diseño, implementación, evaluación.

Una metodología simplificada

Para aplicaciones pequeñas se puede aplicar una metodología em cascada que integra todo el proceso de desarrollo.

1. Identificación del problema:
   * Debemos determinar si el problema es adecuado (¿Hay una solución algorítmica?, ¿Disponemos de fuentes de conocimiento?, ¿El tamaño/objetivo/ complejidad del problema es adecuado? ...).
   * Buscar y evaluar las fuentes de conocimiento.
   * Determinar el conocimiento necesario para el sistema.
   * Establecer los objetivos del sistema (¿Qué respuesta esperamos?).
2. Conceptualización:

Esta fase nos debería dar la perspectiva del problema desde el punto de vista del experto.

* + Deberemos:
    - Detallar los elementos del domino -> descripción informal de la ontología.
    - Descomponer el problema en subproblemas mediante refinamientos sucesivos, descubriendo los bloques de razonamiento.
    - Detallar el flujo de razonamiento y las entradas y salidas de cada subproblema.
    - Detallar y distinguir entre evidencias, hipótesis y acciones y descubrir sus relaciones.
  + Toda esta información la obtendremos a partir de la interacción con el experto (entrevistas) y las fuentes de conocimiento.
  + El resultado será un modelo semiformal del dominio y de los problemas y métodos de resolución.

1. Formalización:

Esta fase transformará la perspectiva del experto en la perspectiva del ingeniero del conocimiento.

* + Decidir el formalismo de representación del conocimiento adecuado.
  + Identificar el espacio de búsqueda.
  + Analizar la tipología de los problemas y bloques de razonamiento y decidir los métodos de resolución de problemas adecuados.
  + Analizar la necesidad de tratamiento de incertidumbre y/o información incompleta.

1. Implementación:
   * Construir una ontología del dominio.
   * Encajar los problemas identificados en las metodologías de resolución de problemas escogidas.
   * Construir los diferentes módulos que correspondan a cada problema siguiendo el conocimiento obtenido.
   * Si utilizamos una aproximación basada en prototipado rápido construiremos el prototipo inicial y lo iremos aumentando incrementalmente.
2. Validación y Prueba:
   * Escoger casos representativos y resolverlos mediante el sistema.
   * Los casos deberían incluir tanto casos usados para la construcción del sistema como casos nuevos.
   * Si seguimos una estrategia de construcción incremental esta fase se irá repitiendo a medida que se desarrolle el prototipo.
   * La validación de SBC es más compleja que la de los sistemas de software habituales.

**Resolución de problemas en SBC**

Tipos de problemas en SBC

La identificación de tipologías de problemas en SBC facilita su desarrollo. Cada tipo permite determinar:

* Un conjunto de tareas usuales.
* Un conjunto de metodologías de resolución específicas.
* Métodos de representación del conocimiento e inferencia adecuados.

Clasificaremos los problemas a resolver en un SBC a partir de dos tareas genéricas:

* Tareas de análisis: interpretación de un sistema.
* Tareas de síntesis: construcción de un sistema.

*Tareas de análisis:*

* Identificación, nos dice que tipo de sistema tenemos:
  + Monitorización, detecta discrepancias de comportamiento.
  + Diagnóstico, explica las discrepancias.
* Predicción, nos dice que salida podemos esperar.
* Control, determina que entradas nos permiten obtener ciertas salidas.

*Tareas de síntesis:*

* Especificación, busca las restricciones que se deben satisfacer.
* Diseño, genera una configuración de elementos según unas restricciones.
* Ensamblaje, construye un sistema juntando elementos.

Cada tipo de problema genérico tiene unas características especiales y existen métodos de resolución adecuados para cada tipo. Nos centraremos en dos métodos:

* Clasificación Heurística.
* Resolución Constructiva.

Clasificación Heurística

Es adecuado para tareas de análisis i el objetivo es escoger una solución de entre un conjunto limitado. Asociamos los datos de entrada con las soluciones (simple o mediante razonamiento).

3 fases:

1. Abstracción de datos (Definicional, cualitativa, generalización, …).
2. Asociación heurística.
3. Refinamiento.

Diagrama

Descripción generada automáticamente

*Adquisición del conocimiento*

La adquisición del conocimiento que permite resolver un problema mediante clasificación heurística se puede hacer de forma sistemática. Podremos distinguir tres conjuntos de conceptos:

* Solución abstracta: son arquetipos de solución, soluciones posibles a nuestro problema.
* Problema abstracto: características que definen arquetipos de problema, útiles para seleccionar los arquetipos de solución.
* Problema concreto: selección de datos de entrada del problema que permiten conectar con arquetipos de problema.

Entre cada conjunto de conceptos deberemos extraer el conjunto de deducciones que llevan del uno al otro.

* De los datos del problema al arquetipo de problema: reglas de abstracción.
* De los arquetipos de problema a arquetipos de solución: reglas de asociación heurística.

Para cada grupo de reglas deberemos:

* Obtener qué conceptos del primer conjunto (antecedentes) están asociados con los del segundo (consecuentes).
* Escoger como antecedentes de las reglas los conceptos que sean diferenciadores de cada consecuente (separabilidad).
* Si es necesario se deberían añadir conceptos intermedios que lleven de los antecedentes a los consecuentes y crear las cadenas de deducción necesarias.
* Observar la confianza de las asociaciones entre antecedentes y consecuentes (incertidumbre).

Si las soluciones abstractas no son aplicables directamente -> determinar las reglas de refinamiento.

El objetivo de la clasificación heurística es buscar cadenas de deducción que permitan ir desde la entrada (o la interacción con el usuario) hasta tomar la decisión sobre los créditos. Será importante apuntar hacia la ‘sencillez’, no estamos buscando fórmulas matemáticas sobre la ‘puntuación’ de una petición para decidir, si no abstraer ‘qué cosas se usan para decidir’ y ‘cómo’.

*Pasos de la clasificación heurística*

PASO 1: Vocabulario de soluciones

Para empezar a atacar el problema, tenemos que entender qué palabras clave usaríamos para describir un ‘arquetipo de solución’. (p. ej., no conceder/conceder con rebaja/conceder con aumento)

PASO 2: Vocabulario de problemas y asociación heurística

Intentamos buscar ‘reglas informales’ para distinguir en qué caso estaríamos de lo anterior. En el proceso, identificamos y abstraemos características del problema abstracto, posiblemente agrupando diferentes conceptos. Esta sería la parte ‘informal’ de la asociación heurística.

(p. ej., ¿cuándo conceder con rebaja? O ¿cuándo no conceder?).

Intentamos categorizar los parámetros utilizados, abstrayendo y agrupando de forma que tenga sentido. Al final del proceso, la casuística de las reglas debería poder cubrir toda posibilidad, con lo cual un número muy elevado de categorías (cada una con diferentes posibles valores) puede llevarnos a necesitar de un conjunto enorme de reglas.

(p. ej., morosidad o cheques sin fondo -> fiabilidad buena / mala / muy mala).

PASO 3: Formalización de asociación heurística

Intentamos formalizar las ‘reglas’ de asociación heurística. El conjunto de reglas debería poder cubrir toda casuística. Deberíamos aprovecharnos de que no necesitamos saber todo del problema concreto. Podemos usar prominencia (salience) en las reglas (bueno, normal o malo). Crear el conjunto potencia de toda posible entrada (para toda categoría y valor, su solución abstracta) es poco extensible y explicable.

(p. ej., si apoyo financiero = regular y bienes = malo entonces no conceder)

PASO 4: Abstracción

Generamos el conjunto de reglas que nos permiten transformar la entrada del problema al vocabulario utilizado para la asociación heurística. El conjunto de datos de entrada solo se usará para crear el problema concreto y para refinar la solución concreta (lo que usamos aquí no se usa para la asociación, debería quedar abstraído en el apartado anterior).

(p. ej., si suma bienes > 2M entonces bienes = bueno / si suma bienes < 1M entonces bienes = malo).

PASO 5: Refinamiento

Para dos de nuestros casos, requerimos saber más que lo que hay en el problema concreto para decidir mejor. Debería poder cubrir toda la casuística, se puede usar valores concretos (como el número exacto del valor del patrimonio en euros).

(p. ej., si conceder con interés preferente y cliente es hermano del director entonces rebaja interés: 2%).

Ejemplo completo:

Diagrama

Descripción generada automáticamente

Las reglas de abstracción normalmente van de A a B. A veces tenemos reglas de A a A: permiten resumir un conjunto de condiciones en un solo predicado (que luego se usa en el antecedente de otras reglas).

Las reglas de asociación heurística normalmente van de B a C. No puede incluir predicados o datos de A. De nuevo, podemos tener reglas de B a B para generar predicados usados en otras reglas.

Las reglas de refinamiento normalmente van de C a D, utilizando también predicados y datos de A. No se deben utilizar predicados de B. Se pueden incluir funciones o procedimientos con elementos de C y A como parámetros.

PASO 6: Verificación

Una vez el proceso anterior está completado, se debe poner un esfuerzo en responder a las siguientes preguntas:

* ¿Los conjuntos de reglas de cada fase son exhaustivos de por sí? Si no, es posible que el programa no sepa cómo categorizar una entrada concreta.
* Si no, ¿los conjuntos de reglas en su totalidad son exhaustivos? Por ejemplo, si concedemos con interés preferente, las reglas de refinamiento no tienen por qué ser exhaustivas para casos con fiabilidad mala porque nunca deberíamos llegar a ese caso.
* ¿Las reglas son coherentes? ¿Tiene cada regla sentido semántico al analizarla individualmente? Al formalizar la clasificación e intentar alcanzar la cobertura total de casos es posible que aparezcan reglas con poco sentido, que tal vez deberían ser descompuestas en varias reglas coherentes.

PASO 7: Extensión

Para tratar con todo este proceso, la construcción del sistema de reglas puede ser creada iterativamente (agile) a partir de un prototipo vertical, añadiendo nuevos conceptos paulativamente (como hacer una primera fase con solo fiabilidad, y luego ir reescribiendo reglas para incluir el análisis de los bienes o el compromiso…).

Se debe notar, sin embargo, que pasos como la abstracción del problema concreto puede generar agrupaciones o simplificaciones que dificultarían la extensión de un prototipo si no se hacen pronto.

Resolución constructiva

Orientada a problemas en los que no existe un número acotado de soluciones. La resolución implica construir la solución a partir de un conjunto de elementos (acciones, componentes, fallos, …). Se aplica en problemas de síntesis y los métodos de búsqueda son aplicables, pero con un coste impracticable.

La construcción de la solución necesita conocimiento sobre:

* El modelo de la estructura de la solución.
* El modelo del comportamiento de los elementos de la solución.
* Los operadores que permiten construir la solución.
* El conjunto de restricciones entre los elementos y la solución.
* La forma de evaluar las decisiones sobre la construcción de la solución y sobre la solución total o parcial.

Las restricciones pueden ser:

* Sobre la configuración de los componentes (físicas, temporales, …).
* Respecto a las entradas/salidas/precondiciones/postcondiciones de los operadores de construcción.

Métodos de resolución constructiva:

* Proponer y aplicar: partimos desde una solución vacía. Seleccionamos el operador de entre los posibles que nos permita extender la solución actual hasta llegar a obtener la solución completa.
* Mínimo compromiso: partimos desde una solución inicial completa. Seleccionamos el operador que nos permite extender la solución actual que menos restricciones nos imponga a las decisiones futura.

*Proponer y aplicar:*

* Buscamos en el espacio de soluciones parciales.
* Partimos de una solución inicial vacía o una solución incompleta.
* Cada paso va completando la solución.
* Siempre elegimos el mejor operador.
* Nos mantenemos en el espacio de soluciones.
* Necesitamos conocimiento exhaustivo sobre:
  + Operadores de resolución del problema.
  + Restricciones y relaciones entre los componentes de la solución.
  + Evaluación del efecto de los operadores en la solución.
  + Evaluación de la bondad de la solución.
* Podemos plantear la resolución de diferentes maneras:
  + Construcción secuencial (necesita mucho conocimiento para ser eficiente).
  + Descomposición jerárquica de tareas (más eficiente, pero requiere obtener operadores de descomposición).

Proceso de resolución:

1. Inicializar el objetivo: se crea el elemento que define la solución actual.
2. Proponer un operador: se seleccionan operaciones plausibles sobre la solución actual.
3. Podar operadores: se eliminan operadores de acuerdo con criterios globales.
4. Evaluar operadores: se comparan los efectos de los operadores sobre la solución y se evalúa su resultado.
5. Seleccionar un operador: se escoge el operador mejor evaluado. Si no hay operadores adecuados se reconsideran pasos anteriores.
6. Aplicar el operador: se aplica el operador al estado actual.
7. Evaluar el objetivo: se para si se ha llegado al objetivo final o se reinicia el proceso.

*Mínimo Compromiso:*

* Exploramos en el espacio de soluciones completas.
* Partimos de una solución inicial completa (también puede ser no solución).
* Vamos modificando la solución mejorándola o corrigiéndola.
* La elección del operador a aplicar la define la estrategia de mínimo compromiso\_ mínima modificación que imponga menos restricciones futuras.
* Permitimos pasar entre el espacio de soluciones y no soluciones.

Proceso de resolución:

1. Partir de una solución inicial no óptima, si es posible, que satisfaga las restricciones.
2. Hacer una modificación sobre la solución usando la heurística de mínimo compromiso (acción que menos restricciones imponga sobre la solución).
3. Si la modificación viola alguna de las restricciones deshacer alguno de los pasos anteriores haciendo las mínimas modificaciones (puede no ser el último paso).

Ej.: Queremos planificar la mejor trayectoria de un robot en una habitación que tiene un conjunto de obstáculos que queremos evitar. Operadores: mover hacia delante o hacia atrás a cierta velocidad cierta distancia y girar cierto número de grados.

* Restricciones globales: llegar a la puerta de salida, trayectoria mínima en recorrido y tiempo.
* Restricciones de elección de operadores: no chocar con obstáculos o la pared, mantener la distancia para poder maniobrar.
* Evaluación de los operadores:
  + Mover: mejor cuanto más lejos y más deprisa nos lleve al objetivo.
  + Girar: mejor cuanto más lejos deje los obstáculos de nuestra trayectoria.

**Razonamiento con incertidumbre**

Todos los mecanismos de representación de conocimiento vistos están basados en la lógica bajo estos supuestos:

* Todo hecho sobre el que razonemos debe poder ser evaluado como cierto o falso.
* Para poder razonar necesitamos tener todos los hechos a nuestra disposición.

Pero en la práctica, nos encontramos con estos problemas:

* Representar el conocimiento para cubrir todos los hechos que son relevantes para un problema es difícil.
* Existen dominios en los que se desconocen todos los hechos y reglas necesarias para resolver el problema.
* Existen problemas en los que aún teniendo las reglas para resolverlos no disponemos de toda la información necesaria o no tenemos confianza absoluta en ellas.
* En otros problemas las reglas no se aplican siempre o su confianza cambia con la confianza que tenemos en los hechos.

Modelos probabilistas

Los modelos probabilistas se basan en la teoría de la probabilidad. Las probabilidades se utilizan para modelizar nuestra creencia sobre los posibles valores que pueden tomar los hechos.

Cada hecho tendrá una distribución de probabilidad asociada que nos permitirá tomar decisiones. La probabilidad de un hecho podrá ser modificada por nuestra creencia en otros hechos que estén relacionados.

El elemento básico de teoría de probabilidades es la variable aleatoria. Una variable aleatoria tiene un dominio de valores, podemos tener variables aleatorias booleanas, discretas o continuas.

Definiremos una proposición lógica como cualquier fórmula en lógica de enunciados o predicados. Una proposición lógica tendrá asociada una variable aleatoria que indicará nuestro grado de creencia en ella.

Una variable aleatoria tendrá asociada una distribución de probabilidad. La forma de expresar esta distribución de probabilidad dependerá del tipo de variable aleatoria (discretas: binomial, multinomial, …, Continuas: normal, X2, …). Nosotros trabajaremos solo con variables aleatorias discretas. La unión de variables aleatorias se puede describir mediante una distribución de probabilidad conjunta.

Denotaremos como P(a) la probabilidad de que la proposición (variable aleatoria) A tenga el valor a. Por ejemplo, la proposición Fumar puede tener los valores {fumar, ¬fumar}, P(¬fumar) es la probabilidad de la proposición Fumar = ¬fumar.

Denotaremos como P(A) al vector de probabilidades de todos los posibles valores de la proposición A.

Definiremos como probabilidad a priori (P(a)) asociada a una proposición como el grado de creencia en ella a falta de otra información.

Definiremos como probabilidad a posteriori o condicional (P(a|b)) como el grado de creencia en una proposición tras la observación de proposiciones asociadas a ella.

La probabilidad a posterior se puede definir a partir de probabilidades a priori como:

P(a|b) = P(a ∧ b) / P(b)

Esta fórmula transforma en la regla del producto:

P(a ∧ b) = P(a|b) \* P(b) = P(b|a) \* P(a)

*Axiomas de probabilidad:*

Serán el marco que restringirá las cosas que podremos creer y deducir:

* Toda probabilidad está en el intervalo [0, 1]

0 <= P(a) <= 1

* La proposición cierto tiene probabilidad 1 y la proposición falso tiene probabilidad 0.

P(cierto) = 1 P(falso) = 0

* La probabilidad de la disyunción se obtiene mediante la fórmula:

P(a ∨ b) = P(a) + P(b) − P(a ∧ b)

*Inferencia probabilística*

Marginalización: probabilidad de una proposición atómica con independencia de los valores del resto de proposiciones.



Probabilidades condicionales: probabilidad de una proposición dados unos valores para algunas proposiciones e independiente del resto de proposiciones (a partir de la regla del producto).



El valor α es un factor de normalización que corresponde a factores comunes que hacen que las probabilidades sumen 1.

*Problema*

Hacer estos procesos de inferencia requiere almacenar y recorrer la distribución de probabilidad conjunta de todas las proposiciones. Suponiendo proposiciones binarias, el coste en tiempo y espacio es O(2n) siendo n el número de proposiciones.

Para cualquier problema real, estas condiciones son impracticables. Necesitamos mecanismos que nos simplifiquen el coste del razonamiento.

*Independencia probabilística*

Por lo general, no todas las proposiciones que aparecen en un problema están relacionadas entre sí. Muestran la propiedad que denominaremos independencia probabilística. Esto quiere decir que unas proposiciones no influyen en las otras y, por lo tanto, podemos reescribir sus probabilidades como:

P(X|Y) = P(X); P(Y|X) = P(Y); P(X, Y) = P(X) P(Y)

Dadas estas propiedades podremos reescribir las probabilidades conjuntas de manera más compacta reduciendo la complejidad.

*Regla de Bayes*

Regla del producto:

P(X,Y) = P(X|Y) P(Y) = P(Y|X) P(X)

Esto nos lleva a lo que denominaremos la regla de Bayes:

Texto

Descripción generada automáticamente

Esta regla y propiedad de independencia serán el fundamento del razonamiento probabilístico y nos permitirá relacionar las probabilidades de unas evidencias con otras.

Suponiendo que podemos estimar exhaustivamente todas las probabilidades que involucran todos los valores de la variable Y, podemos reescribir la fórmula de Bayes como:

P(Y|X) = αP(X|Y) P(Y)

Suponiendo independencia condicional entre dos variables podremos escribir:

P(X, Y|Z) = P(X|Z) P(Y|Z)

De manera que:

P(Z|X, Y) = α P(X, Y|Z) P(Z) = α P(X|Z) P(Y|Z) P(Z)