INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL

Códigos: INFORMÁTICA DE SISTEMAS **Código carrera: 40** **Código asignatura: 209**

Junio 2000-2001, **2ª Semana**, **DURACIÓN: 2 HORAS,** **Material permitido: NINGUNO**

Importante: Ponga el nombre en todas las hojas. No sólo se valorará que el resultado sea correcto, sino también la claridad en la exposición de los pasos que se han seguido en la resolución, que el examen esté compensado y que no incluya errores conceptuales importantes.

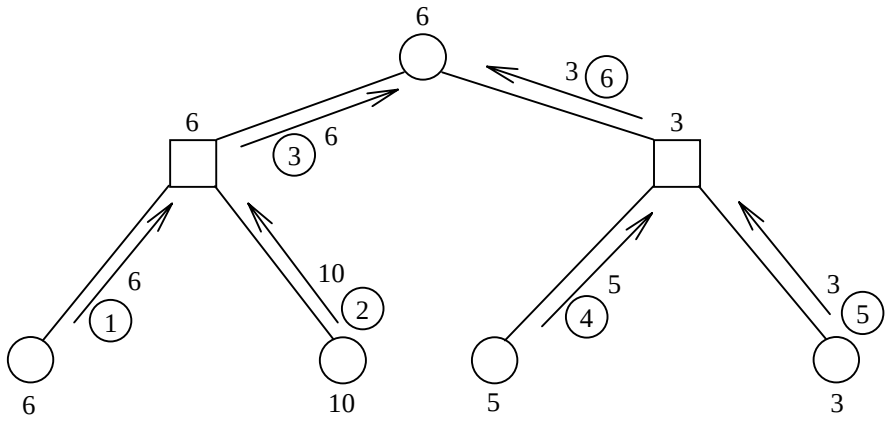
1. (Valoración: *2 puntos***)**

¿En qué se asemejan y en qué se diferencian el método MINIMAX y el Método de poda alfa-beta? Describa en detalle un ejemplo en el que dichas diferencias queden reflejadas.

----------SOLUCIÓN----------

Los métodos MINIMAX y de poda alfa-beta se utilizan en problemas de búsqueda con dos adversarios cuyo objetivo es ganar una partida en la que realizan movimientos alternativos. Normalmente, en cualquiera de estos dos métodos se genera un árbol de búsqueda con una profundidad limitada a partir de la situación inicial de la partida. Cada nivel de este árbol está asociado a un turno de movimiento para uno de los dos jugadores; lógicamente, se supone que cada jugador elegirá siempre aquel movimiento que más ventaja le dé en la partida. A los nodos hoja del árbol de búsqueda se les asocia el valor de una función de evaluación heurística (*fev*) que es una estimación del tipo de juego que cuelga de ese nodo hoja.

El método MINIMAX explora exhaustivamente el árbol de búsqueda mediante un método de búsqueda en profundidad. Considérese el siguiente ejemplo, donde se representa mediante círculos el turno del jugador que tiene que mover en la situación actual de la partida y mediante rectángulos el turno de su contrincante. Debajo de cada nodo hoja aparece un número que indica lo prometedora que es esa situación de la partida para que gane el jugador que tiene que mover en la situación actual (nodo raíz del árbol de búsqueda). Al lado de cada arco se especifica el valor de la jugada alcanzado si la misma se desarrolla por ese camino y, encerrado en un círculo, el ciclo del algoritmo de búsqueda en profundidad en que se devolvería dicho valor.



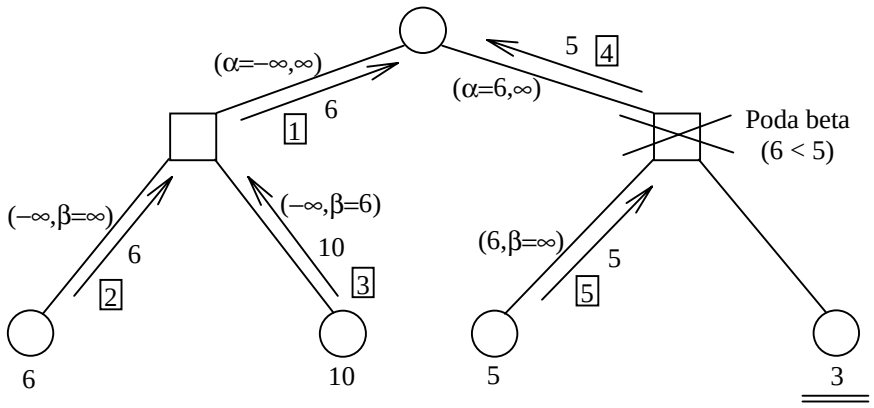
Finalmente, la mejor jugada es la de la izquierda, que conduce a un valor 6, frente a la jugada de la derecha, que conduce a un valor 3.

En el árbol de búsqueda anterior, en el ciclo 3 el jugador que tiene que mover sabe que por la izquierda puede llegar a una jugada de valor 6. En el ciclo 4, sin haber explorado todavía el nodo

hoja etiquetado con 3, sabe que por la derecha podrá llegar a una jugada de valor 5 o menor que 5. Por tanto, siempre elegirá la jugada de la izquierda, sea cual sea el valor de la *fev* asociado al nodo hoja situado más a la derecha. Por tanto, la exploración que el método MINIMAX realiza del nodo hoja etiquetado con 3 es innecesaria.

El método de poda alfa-beta permite corregir situaciones como la anterior en las que el método MINIMAX realiza exploraciones innecesarias de ciertos subárboles del árbol de búsqueda. Para ello, a la vez que se realiza una exploración en profundidad del árbol de búsqueda, existe la posibilidad de realizar podas en el mismo. Una poda alfa se realiza en un nodo desde el que se intente acceder a situaciones de la partida con valores altos de la *fev* y una poda beta desde el tipo contrario de nodos. En cada llamada recursiva desde un nodo a uno de sus nodos hijo, se pasan dos parámetros: alfa y beta. El valor recibido como consecuencia de la llamada recursiva permite actualizar (incrementar o dejar igual) el valor de alfa, si la llamada se hizo desde un nodo desde el que se intente acceder a situaciones de la partida con valores altos de la *fev*; el valor actualizado es beta (se decrementa o se deja igual) si la llamada se hizo desde el tipo contrario de nodos. El valor que se devuelve al nodo padre después de realizada una llamada recursiva a cada nodo hijo es el último valor actualizado de alfa o beta, según corresponda. Al principio, en la primera llamada recursiva desde el nodo raíz, alfa =–∞ y beta=+∞. Se realiza una poda desde un nodo tan pronto como los valores que gestiona de alfa y beta cumplan la siguiente relación: alfa ≥ beta.

El ejemplo anterior quedaría ahora de la siguiente forma, donde el orden de realización de cada llamada recursiva se representa mediante un número encerrado en un rectángulo:



En la llamada recursiva 2 se actualiza el valor de beta de ∞ a 6. Después de la llamada recursiva 3 se devuelve hacia arriba el último valor de beta actualizado: 6. En la llamada recursiva 1 se actualiza el valor de alfa de –∞ a 6. En la llamada recursiva 5 se actualiza el valor de beta de ∞ a 5. En este momento se realiza una poda beta y el nodo etiquetado con "3" queda sin visitar. Como la última vez que se actualiza el valor de alfa en el nodo raíz es por el camino de la izquierda, ésta será la mejor jugada posible para esa situación actual de la partida.

2. (Valoración: *5 puntos***)**

Suponga que en un mundo de objetos se parte de la existencia de los siguientes objetos: una mesa plegable, mesa1, de color marrón; un cubo de color verde, al que llamaremos cubo1, con una densidad de 5 y con una longitud, altura y anchura todas ellas iguales a 10; un segundo cubo de color azul, cubo2, con un peso de 4 que se encuentra inicialmente encima del cubo1.

Además se parte de los siguientes datos.

Un objeto tiene como especialización un objeto físico que puede tener los siguientes atributos: volumen, peso, densidad, encima, debajo, frágil, menos pesado, apilar-encima, color. Todos estos

atributos son del tipo objeto-fisico-slot y por ello tienen las propiedades: dominio (con el valor objeto-físico) y métodos (con la secuencia de valores: herencia, prolog, defecto).
Una mesa plegable es un objeto físico que tiene un atributo peso cuyo valor por defecto es 5.
Un cubo es un objeto físico que tiene las propiedades altura, longitud y anchura.
Los métodos de inferencia de un atributo de un objeto físico son la herencia, las reglas estilo prolog (lógica de predicados) que se pueden definir para calcular su valor y el valor por defecto. Aplicándose en dihco orden hasta que se devuelva un valor, esto es: herencia, prolog y valor-defecto.
Las reglas de inferencia que se conocen a priori para el cálculo de los valores de ciertos atributos son las siguientes:

- Un objeto se considera **frágil** si su peso es menor o igual que 5.
- Un objeto es **menos pesado** que otro si el peso del primero es menor que el peso del segundo.
- Un objeto **se puede apilar** encima de otro si a) el primero es menos pesado que el segundo o si b) el segundo no es frágil.
- El **peso** de un objeto es igual al volumen por la densidad.
- El **volumen** de un objeto es igual a su altura por su longitud y por su anchura.

1. Crear un sistema de marcos y procedimientos que capte el conocimiento encerrado en la descripción dada, concretando todos los objetos y relaciones del dominio, así como los métodos de inferencia. Utilice la siguiente notación para la descripción de las entidades del dominio. **Añada finalmente la representación gráfica** del sistema generado.

```
<marco> ::=          <clase> | <instancia>
<clase>   ::=        clase <nombre-de-clase> es
                            superclase <espec-super>;
                             <atributos>
                                fin
<instancia> ::=      instancia <nombre-de-instancia> es
                       instancia-de <espec-super>;
                        <atributos>
                        fin
<espec-super> ::=  <nombre-de-clase>{,<nombre-de-clase>}* | nil
<atributos> ::=      <par-atributo-faceta>{;<par-atributo-faceta>}* |   <vacío>
<par-atributo-faceta> ::=      <nombre-de-atributo>=(<faceta>{,<faceta>}*)
<faceta>   ::=       <nombre-de-faceta> <valor> |  demonio <tipo-de-demonio>  <llamada-a-demonio>
<nombre-de-faceta> ::=       valor | valor-por-defecto
<tipo-de-demonio>::=         si-se-necesita | si-se-añade | si-se-borra
<valor>    ::=       <constante-elemental> | <nombre-de-instancia>
<vacío>    ::=
```

Los demonios se pueden especificar en cualquier pseudocódigo. Si bien se valorará el uso de la notación simbólica estilo cláusulas prolog siguiente. Se explica esta notación con la definición del método para determinar si un objeto se considera frágil:

```
(FRÁGIL
 (frágil ?x ?valor):-
        (peso ?x ?peso1)
        (eval (<=?peso1 5) ?valor))))
```

Esto es, un objeto se considera frágil si su peso es menor o igual que 5. t representa el valor T o cierto. El interrogante delante de un nombre cualquiera denota cualquier variable. Así ?peso1 es una variable de nombre peso1. Eval es una función que evalúa sus argumentos en secuencia asignando el valor obtenido en su primer argumento a la variable indicada en el segundo. Cuando en una regla aparece una cláusula con un cierto nombre de atributo el sistema busca en la base de conocimientos todas las instancias que tengan dicho atributo. Por ejemplo, (peso ?x ?peso1) se instanciaría creando las asociaciones variable valor siguientes: (?x cubo2) y (?peso1 4).

2. A partir de la definición dada en el punto 1. Se pide describir el proceso de inferencia que determina los objetos apilables sobre la mesa1. Para ello puede indicarse la secuencia de ternas accedidas (objeto atributo valor). Por ejemplo, (cubo2 peso 5).

----------**SOLUCIÓN**----------
-

Representación gráfica

La representación gráfica de los objetos del dominio es la siguiente:

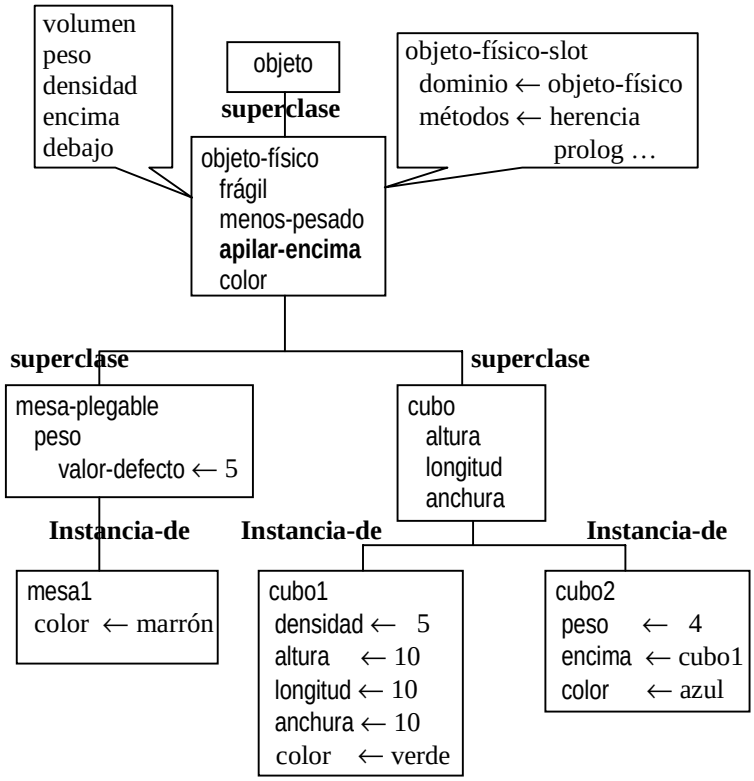


Figura 1 Jerarquía de relaciones entre objetos.

Representación lineal

Para construir la representación lineal se describirán paulatinamente las estructuras propuestas en el enunciado siguiendo la notación indicada y siguiendo un orden de especificidad creciente en dichas descripciones.

Objetos del dominio

clase objeto **es**
    superclase nil;
fin

Debido a las restricciones de la notación propuesta, donde no pueden definirse campos o *slots* de forma aislada y donde no existen las propiedades *generalizaciones* ni *especializaciones* como ocurre en los ejemplos del capítulo 7 del libro de problemas, se propone incluir una descripción genérica de sus *slots* tipo (objeto-físico-slot) además de las descripciones más concretas que cubren cada una de las propiedades del enunciado.
Para que el sistema funcionara conforme a lo indicado en el enunciado, cada vez que se intentara inferir el valor de un *slot* de un objeto físico, en primer lugar se llamaría al procedimiento

APLICA-MÉTODO que a su vez llamaría a cada uno de los métodos más concretos siguiendo la secuencia propuesta.

clase objeto-físico **es**
    superclase objeto;
    objeto-físico-slot=(**demonio si-se-necesita** APLICA-MÉTODO(herencia, prolog, valor-por-defecto));
    volumen=(**demonio si-se-necesita** VOLUMEN(?altura, ?longitud, ?anchura));
    peso=(**demonio si-se-necesita** PESO(?densidad, ?volumen));
    frágil=(**valor** (T NIL), **demonio si-se-necesita** FRÁGIL(?peso1));
    menos-pesado=(**valor** (T NIL), **demonio si-se-necesita** MENOS-PESADO(?peso1, ?peso2));
    apilar-encima=(**valor** (T NIL), **demonio si-se-necesita** APILAR-ENCIMA(?objeto1, ?objeto2));
    volumen=(**demonio si-se-necesita** VOLUMEN(?altura, ?longitud, ?anchura));
    encima=(**demonio-si-se-necesita** ENCIMA(?x ?y);
    debajo=(**demonio-si-se-necesita** DEBAJO(?x ?y));
    densidad=(**valor-por-defecto** 5);
    color=(**valor-por-defecto** marrón)
fin

Nota: se han elegido los valores por defecto 5 y marrón respectivamente para los campos densidad y color para poder así describir el atributo según la notación propuesta.

clase mesa-plegable **es**
    superclase objeto-físico;
    peso=(**valor-por-defecto** 5)
fin

clase cubo **es**
    superclase objeto-físico;
    altura=(**valor-por-defecto** 5);
    longitud=(**valor-por-defecto** 5);
    anchura=(**valor-por-defecto** 5)
fin

Nota: se ha elegido valor por defecto de todos los atributos el valor 5 para poder así describir el atributo según la notación propuesta.

instancia cubo1 **es**
    instancia-de cubo;
    densidad=(**valor** 5);
    altura=(**valor** 10);
    longitud=(**valor** 10);
    anchura=(**valor** 10);
    color=(**valor** verde)
fin

instancia cubo2 **es**
    instancia-de cubo;
    peso=(**valor** 4);
    encima=(**valor** cubo1);
    color=(**valor** azul)
fin

instancia mesa1 **es**

instancia-de mesa-plegable;
color=(**valor** marrón)
fin

Métodos e inferencia

Se definen a continuación los demonios de las propiedades del objeto físico:

El demonio genérico APLICA-MÉTODO se supone siempre presente para cualquier *slot* de un objeto físico y sólo tiene que aplicar en secuencia cada uno de los métodos de inferencia señalados hasta que alguno devuelva un valor o hasta que se devuelva *valor-no-inferido*.

```
(VOLUMEN
(volumen ?x ?vol):-
        (altura ?x ?altura)
        (longitud ?x ?longitud)
        (anchura ?x ?anchura)
        (eval (* ?altura ?longitud ?anchura) ?vol)))

(PESO
(peso ?x ?peso):-
        (densidad ?x ?densidad)
        (volumen ?x ?volumen)
        (eval (* ?volumen ?densidad) ?peso)))

(FRÁGIL
(frágil ?x ?valor):-
        (peso ?x ?peso1)
        (eval (<=?peso1 5) ?valor)))

(MENOS-PESADO
(menos-pesado ?x ?y):-
        (peso ?x ?peso1)
        (peso ?y ?peso2)
        (eval (<=?peso1 ?peso2) t)))

(APILAR-ENCIMA
(apilar-encima ?x ?y):-
        (menos-pesado ?x ?y))
(apilar-encima ?x ?y):-
        (not (fragil ?y))))

(ENCIMA
(encima ?x ?y):-
        (debajo ?y ?x)))

(DEBAJO
(debajo ?x ?y):-
        (encima ?y ?x)))
```

Proceso de inferencia

Tal y como se pide en el enunciado, se presenta un esquema con la secuencia de ternas accedidas (objeto atributo valor) para inferir los objetos apilables sobre la mesa1.
Supongamos que la función que se invoca para preguntar por el valor de un *slot* de un objeto es `dame-valor` y que, de acuerdo con el problema, para poder aplicar los métodos definidos anteriormente mediante cláusulas prolog se utiliza el segundo método de inferencia aplicable, `prolog`.
Recordemos que la secuencia es `herencia`, `prolog` y `valor-por-defecto`. De ahí, por ejemplo, que al intentar inferir el valor del `volumen` de la `mesa1` se intente acceder al valor del `volumen` de la mesa-plegable. En este caso se está aplicando el primer método disponible, `herencia`. Como puede apreciarse en la secuencia, se insiste en la `herencia` hasta llegar a objeto físico. Una vez se ha terminado con dicho camino de inferencia se puede ya acceder al valor del *slot* `peso` del objeto `mesa-plegable`, que es precisamente la superclase de `mesa1`, también accedida en primera instancia al aplicarse el método `herencia`.

```
> (dame-valor apilar-encima mesa1)
  >2 dar (mesa1 apilar-encima prolog)
    >4 dar (mesa1 menos-pesados prolog)
      >6 dar (mesa1 peso prolog)
        >8 dar (mesa1 volumen prolog)
        <8 dar (*infer-sin-valor* (((mesa1 volumen prolog))))
          >10 dar (mesa-plegable volumen prolog)
          <10 dar (*infer-sin-valor*
                        (((mesa-plegable volumen prolog))))
            >12 dar (objeto-fisico volumen prolog)
            <12 dar (*infer-sin-valor*
                        (((objeto-fisico volumen prolog))))
      <6 dar (*infer-sin-valor* (((mesa1 peso prolog))))
        >8 dar (mesa-plegable peso prolog)
        <8 dar (*infer-sin-valor*
                        (((mesa-plegable peso prolog))))
          >10 dar (objeto-fisico peso prolog)
          <10 dar (*infer-sin-valor*
                        (((objeto-fisico peso prolog))))
        >8 dar (mesa-plegable peso valor-por-defecto)
        <8 dar (5 (((mesa-plegable peso valor-por-defecto))))
      >6 dar (cubo2 peso prolog)
      <6 dar (4 (((cubo2 peso prolog))))
      >6 dar (cubo1 peso prolog)
      <6 dar (5000 (((cubo1 peso prolog))))
     <4 dar ((cubo2) (((mesa1 menos-pesados prolog))))
  <2 dar ((cubo2) (((mesa1 apilar-encima prolog))))
(cubo2)
```

Nota sobre la evaluación de este apartado:
Aunque las ternas propuestas se corresponden con un proceso real ejecutado con un sistema concreto descrito en el capítulo 7 del libro de problemas, la notación que se siga no tiene por qué coincidir con la propuesta pero tiene que quedar claramente reflejado el acceso a los *slots* significativos que determinan el camino de inferencia seguido.

3. (Valoración: *3 puntos***)**
Realice un estudio comparativo de los siguientes métodos de representación de conocimiento: *Lógicas no monótonas* y *Reglas*. Haga especial énfasis en los siguientes aspectos:

a) Tipo de conocimiento que permiten modelar.
b) Tipo de inferencias que permiten realizar

Aclaración: las cuestiones teóricas están descritas en los apartados correspondientes de la bibliografía básica de esta asignatura.