
Ejercicio 1 [2.5 puntos]

El orden de los operadores en un problema de búsqueda puede afectar a la resolución de dicho problema con los algoritmos estudiados en la asignatura. Responder razonadamente a las siguientes cuestiones:

1. Es posible encontrar un problema de búsqueda con ramificación 3 y solución minimal a profundidad 3 tal que el orden de los operadores haga que, al utilizar el algoritmo de búsqueda en anchura, el número de nodos analizados sea inferior a 9.

2. En un problema de búsqueda con ramificación r y solución minimal a profundidad p ($p > 0$), ¿cúal es el número mínimo de nodos que tiene que analizar el algoritmo de búsqueda en anchura hasta llegar a la solución?.

3. Dado cualquier problema de búsqueda con ramificación r y solución minimal a profundidad p ($p > 0$), ¿es posible ordenar sus operadores de manera que el algoritmo de búsqueda en anchura sólo tenga que analizar la cantidad mínima de nodos?.

4. Es posible encontrar un problema de búsqueda con ramificación 3 y solución minimal a profundidad 3 tal que el orden de los operadores haga que, al utilizar el algoritmo de búsqueda en profundidad, el número de nodos analizados sea inferior a 9.

5. En un problema de búsqueda con ramificación r y solución minimal a profundidad p ($p > 0$), ¿cúal es el número mínimo de nodos que tiene que analizar el algoritmo de búsqueda en profundidad hasta llegar a la solución?.

6. Dado cualquier problema de búsqueda con ramificación r y solución minimal a profundidad p ($p > 0$), ¿es posible ordenar sus operadores de manera que el algoritmo de búsqueda en profundidad sólo tenga que analizar la cantidad mínima de nodos?.

7. Es posible que el orden de los operadores afecte al número de nodos analizados en el algoritmo primero el mejor.

. . . .

Solución:

Apartado 1.1

No. Si la ramificación es r (3), y la profundidad de la solución minimal es p (3), entonces el algoritmo de búsqueda en anchura tendrá que analizar todos los nodos desde el nivel 0, hasta el nivel $p-1$ ($p \geq 1$), es decir, al menos se han de analizar $r^0 + r^1 + r^2 + ... + r^{(p-1)}$ nodos, y esta cantidad nunca será inferior a $r^{(p-1)}$ (9).

Apartado 1.2

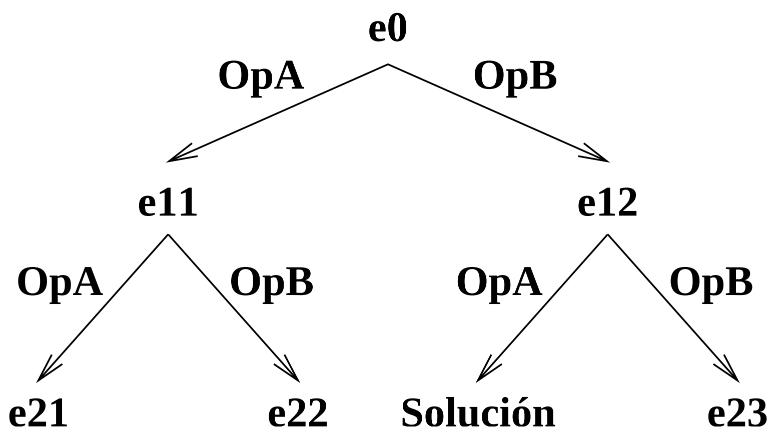
Como se ha explicado antes, el algoritmo de búsqueda en anchura tendrá que analizar todos los nodos desde el nivel 0, hasta el nivel $p-1$. Además de estos tendrá que analizar al menos un nodo del nivel p, la solución. La cantidad mínima se alcanza cuando la solución es el primer nodo generado a profundidad p y, en este caso, el número total de nodos analizados es:

$$r^0 + r^1 + r^2 + ... + r^{(p-1)} + 1 = \sum_{i=0}^{p-1} r^i + 1$$

Apartado 1.3

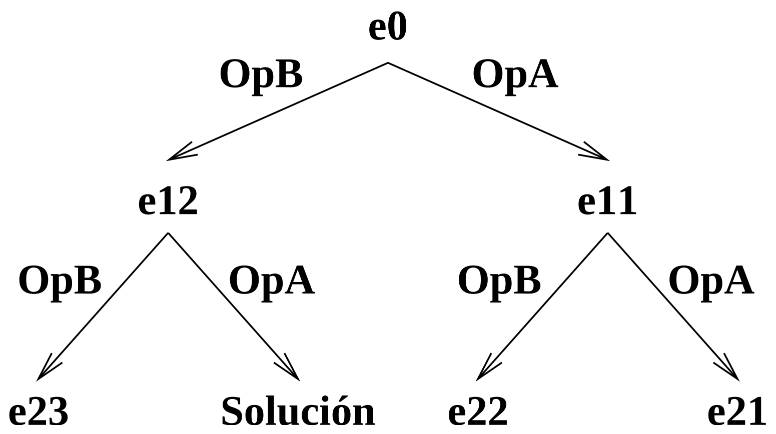
No. Consideremos el siguiente ejemplo con $r = 2$ y $p = 2$. Los operadores son OpA y OpB, y sólo se pueden ordenar de dos maneras:

1. Con operadores = $\{OpA\ OpB\}$, el árbol de búsqueda es:



   y el número de nodos analizados es 6 ($\neq \sum_{i=0}^{p-1} r^i + 1$).

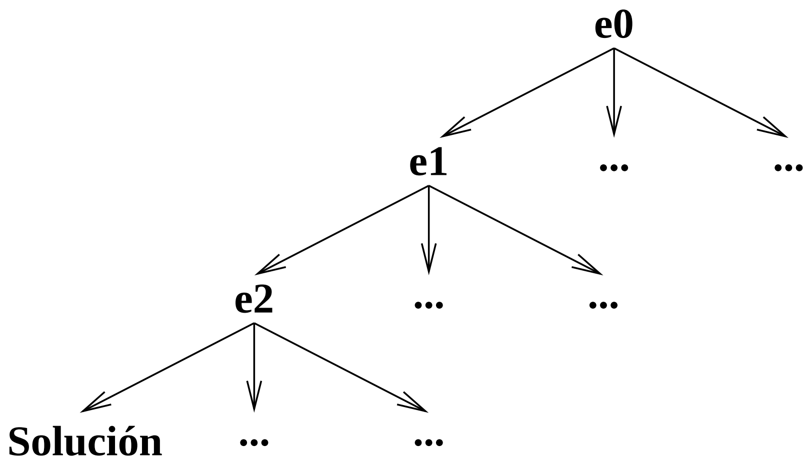
2. Con operadores = $\{OpB\ OpA\}$, el árbol de búsqueda es:



y el número de nodos analizados es 5 ($\neq \sum_{i=0}^{p-1} r^i + 1$).

Apartado 1.4

Sí. Un ejemplo es el siguiente, en el que la solución se encuentra en el primer nodo generado a profundidad 3. El número de nodos analizados es 4.



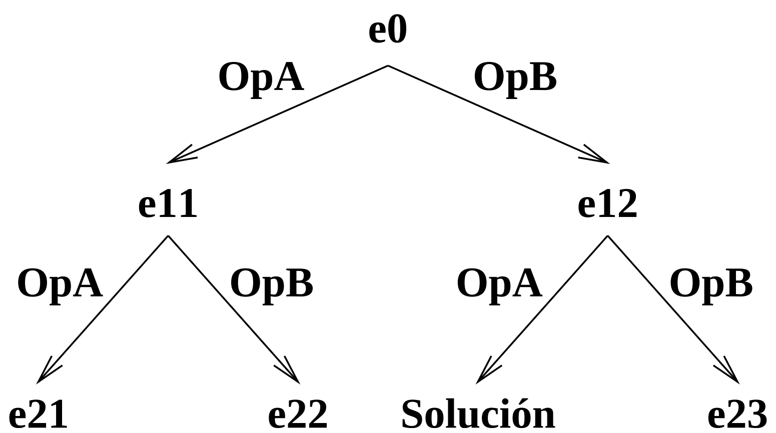
Apartado 1.5

El algoritmo de búsqueda en profundidad analizará una cantidad mínima de nodos si la solución se encuentra en el primer nodo generado a profundidad p, y para llegar a éste bastará con analizar $p + 1$ nodos.

Apartado 1.6

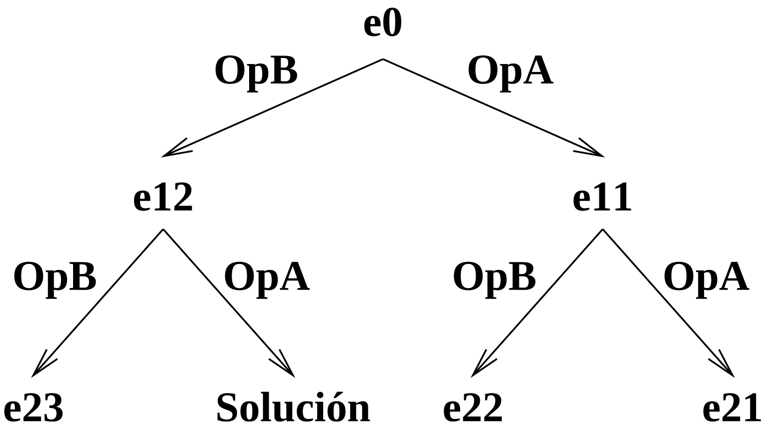
No. Consideremos el siguiente ejemplo con $r = 2$ y $p = 2$. Los operadores son OpA y OpB, y sólo se pueden ordenar de dos maneras:

1. Con operadores $= \{OpA\ OpB\}$, el árbol de búsqueda es:



y el número de nodos analizados es 6 ($\neq p + 1$).

2. Con operadores $= \{OpB\ OpA\}$, el árbol de búsqueda es:



y el número de nodos analizados es 4 ($\neq p + 1$).

Apartado 1.7

En general no, salvo que en el problema haya un estado e y dos operadores $Op1$ y $Op2$ tales que, al aplicar dichos operadores al estado, se obtengan dos sucesores $e1$ y $e2$ con la misma heurística.

Ejercicio 2 [2.5 puntos]

Se considera la siguiente definición del procedimiento de búsqueda A*

```
(defun busqueda-a-estrella-1 ()
  (let ((abiertos (list (crea-nodo-hc
                         :estado *estado-inicial*
                         :camino nil
                         :coste-camino 0
                         :coste-mas-heuristica
                         (heuristica *estado-inicial*))))
        (cerrados nil)
        (actual nil)
        (sucesores nil))
    (loop until (null abiertos) do
      (setf actual   (first abiertos)
            cerrados (cons actual cerrados)
            abiertos (rest abiertos))
      (cond ((es-estado-final actual)
             (return actual))
            (t (setf sucesores (sucesores actual)
                     abiertos (append abiertos sucesores)))))))
```

Explicar si la definición anterior es correcta o incorrecta y, en el caso de ser incorrecta, explicar cuales son los errores y cómo corregirlos para que la definición sea correcta.

. . . .

Solución:

Los errores son:

1. La comprobación de que el nodo `actual` es final es incorrecta. Hay que sustituir `(es-estado-final actual)` por `(es-estado-final (estado actual))`.

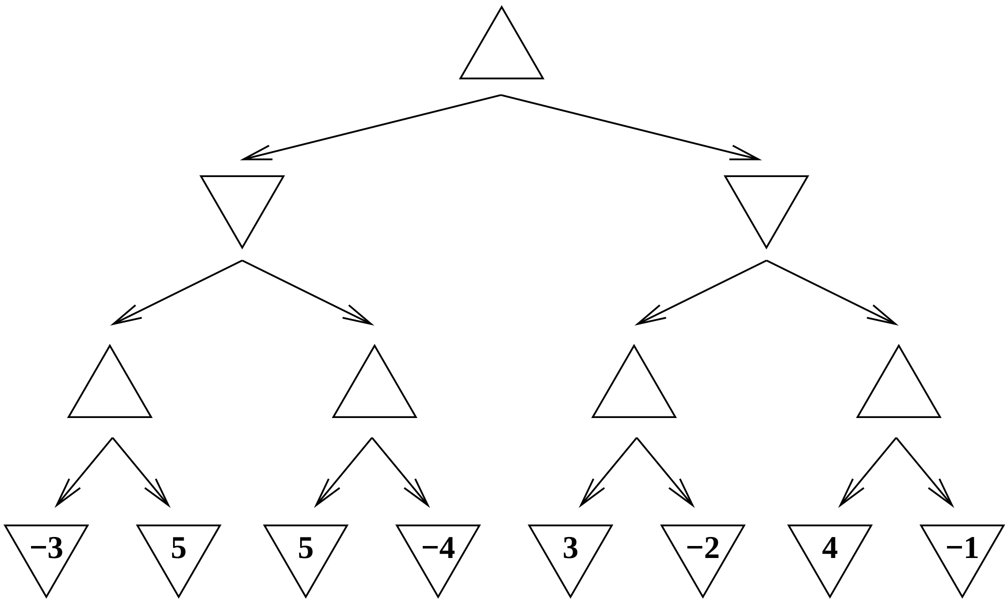
2. El cálculo de los sucesores del nodo `actual` es incorrecto, no se eliminan los repetidos. Hay que sustituir `(sucesores actual)` por `(nuevos-o-mejores-sucesores actual abiertos cerrados)`

3. No se ordena la lista de `abiertos` con respecto a la suma de coste y heurística. Hay que sustituir `(append abiertos sucesores)` por `(ordena-por-coste-mas-heuristica (append abiertos sucesores))`

Ejercicio 3 [2.5 puntos]

Realizar los siguientes apartados:

1. Aplicar la estrategia alfa-beta con cotas iniciales -5 (α) y 5 (β) al siguiente árbol:



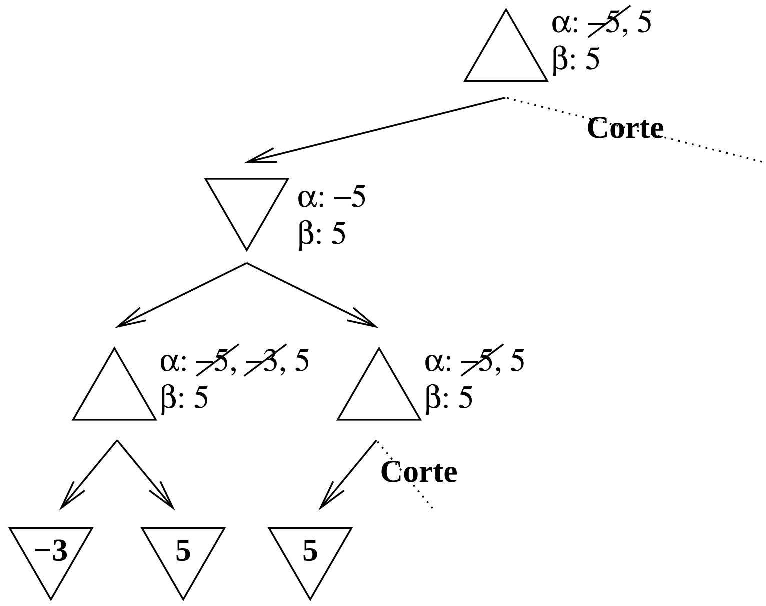
2. En qué condiciones se puede realizar una poda en el primer nivel del árbol, cuando las cotas iniciales son números reales M (α) y N (β), con $M < N$.

3. Aplicar la estrategia alfa-beta con cotas iniciales $-\infty$ (α) y $+\infty$ (β) al árbol del apartado 1.

4. En qué condiciones se puede realizar una poda en el primer nivel del árbol, cuando las cotas iniciales son $-\infty$ (α) y $+\infty$ (β).

Solución:

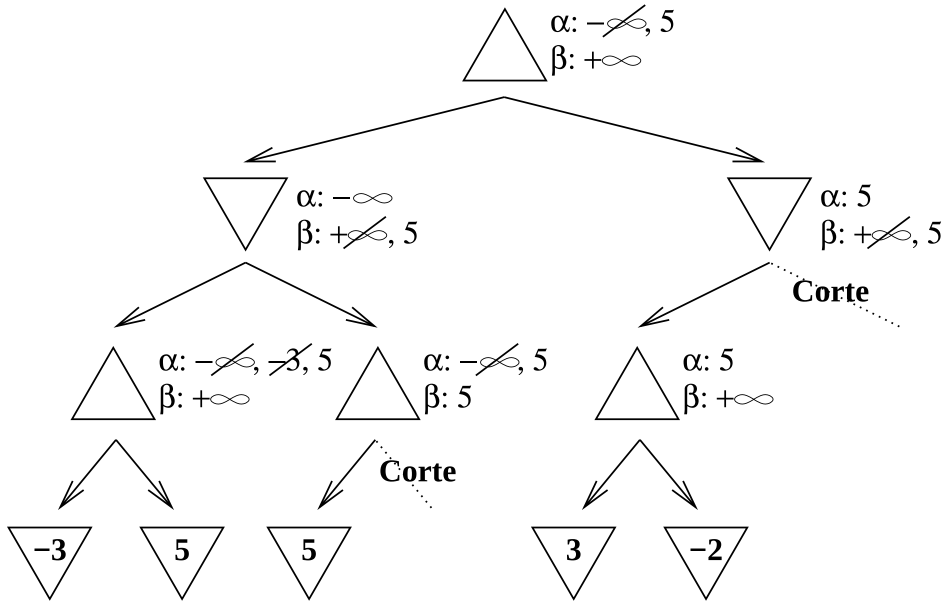
Apartado 3.1



Apartado 3.2

Si el nodo de partida es MAX, cuando el valor obtenido para alguno de sus hijos, distinto del último, sea igual a N. Si el nodo de partida es MIN, cuando el valor obtenido para alguno de sus hijos, distinto del último, sea igual a M.

Apartado 3.3



Apartado 3.4

Nunca. Si el nodo de partida es MAX, el valor obtenido para alguno de los hijos nunca superará o igualará al valor de β del nodo de partida ($+\infty$). Si el nodo de partida es MIN, el valor obtenido para alguno de los hijos nunca será inferior o igualará al valor de α del nodo de partida ($-\infty$).

Ejercicio 4 [2.5 puntos]
Se considera el siguiente programa

```
long(nil,0).
long(cons(X,A),s(N)) :-
   long(A,N).

conc(nil,A,A).
conc(cons(X,A),B,cons(X,C)) :-
   conc(A,B,C).
```

Construir el árbol de resolución SLD correspondiente a dicho programa y a la pregunta

```
?- conc(cons(a,nil),cons(b,nil),A), long(A,N).
```

indicando las respuestas obtenidas.

. . . .

Solución:
El árbol de resolución SLD pedido es el siguiente:

conc(cons(a,nil),cons(b,nil),A0), long(A0,N0).

conc(cons(X1,A1),B1,cons(X1,C1)) :− conc(A1,B1,C1).
{X1/a,A1/nil,B1/cons(b,nil),A0/cons(a,C1)}

conc(nil,cons(b,nil),C1), long(cons(a,C1),N0).

conc(nil,A2,A2).
{A2/cons(b,nil),C1/cons(b,nil)}

long(cons(a,cons(b,nil)),N0).

long(cons(X3,A3),s(N3)) :− long(A3,N3).
{X3/a,A3/cons(b,nil),N0/s(N3)}

long(cons(b,nil),N3).

long(cons(X4,A4),s(N4)) :− long(A4,N4).
{X4/b,A4/nil,N3/s(N4)}

long(nil,N4).

long(nil,0).
{N4/0}

$\square$

La solución es:

$$N0 = s(N3) = s(s(N4)) = s(s(0))$$

$$A0 = cons(a, C1) = cons(a, cons(b, nil))$$
