Universitat Oberta de Catalunya
www.uoc.edu

Estudis d'Informàtica i Multimèdia

INTEL·LIGÈNCIA ARTIFICIAL 1
PAC1 – 2009_1 Prova d'Avaluació Continuada

- Per a dubtes i aclariments sobre l'enunciat, adreceu-vos al consultor responsable de la vostra aula.
- Cal lliurar la solució en un fitxer Word, OpenOffice, PDF o RTF fent servir una de les plantilles lliurades conjuntament amb aquest enunciat. Adjunteu el fitxer a un missatge adreçat a la bústia **lliurament d'activitats**.
- El nom del fitxer ha de ser *CognomsNom*_IA1_PAC1 amb l'extensió *.doc* (Word), *.sxw* (OpenOffice), *.pdf* (PDF) o *.rtf* (RTF), segons el format en què feu el lliurament.
- La data límit de lliurament és el: **13 de Octubre** (a les 24 hores).
- **Raoneu la resposta en tots els exercicis. Les respostes sense justificació no rebran puntuació.**

Enunciat:

Una endevinalla consisteix en donats quatre nombres i un resultat determinar les operacions de suma o resta que cal fer sobre els nombres per trobar aquest resultat. Per exemple:

Nombres: 1,4,3,2
Resultat: 0
Operacions: 4-3-2+1

Suposant que volem resoldre l'endevinalla com un problema de cerca respon a les següents preguntes:

1.- Proposa un espai d'estats per resoldre aquest problema.

2.- Segons la proposta anterior quants estats es consideraran?

3.- Es tracta d'una cerca a cegues d'un estat objectiu. Aplicant l'algoritme de cerca en amplada troba la solució a l'endevinalla.

3.1.- Quants operadors tindrem? Quins seran aquests operadors? Com relacionen els operadors els estats descrits més amunt?

3.2.- Doneu la definició de l'estat inicial i la de l'estat objectiu.

3.3.- Feu una representació de l'arbre de cerca.

3.3.a - A quin nivell de profunditat s'ha trobat la solució?

3.3.b - Quants nodes heu generat?

3.3.c – Mostra l'evolució de la cua de pendents i digues la quantitat de memòria necessària (en nombre de nodes). Considera que s'avalua si un node és estat final quan aquest s'introdueix a la llista de pendents i no quan s'agafa per expandir tal i com diuen els mòduls.

3.3.d - Nodes pendents d'expansió en trobar la solució.

4.- Ara aplicant l'algoritme de cerca en profunditat troba la solució a l'endevinalla i analitza l'eficiència d'aquest algoritme comparant-la amb la de l'anterior.

SOLUCIÓ:

1.- Proposa un espai d'estats per resoldre aquest problema.

Un estat vindrà representat per una tupla (s1; s2; s3; s4) en la que $s_i \in [+1,-1,0]$ representa el signe del nombre en la posició i. El valor 0 indica que encara no s'ha assignat cap signe.

2.- Segons la proposta anterior quants estats es consideraran?

El tamany de l'espai d'estat és 3^n , on n és la quantitat de nombres que ens dona el problema, en aquest cas: 3^4 =81. Ara bé, el nombre de nodes a generar en l'algoritme en amplitud depèn dels operadors que es defineixen. Una possible opció seria que a cada nivell i s'assigni un signe al nombre i-èssim. En l'exemple, en el primer nivell assignarem signe al 1, en el segon al 4, en el tercer al 3 i per últim al dos. Per tant, tenim un factor de ramificació de 2, amb una profunditat de 5, amb la qual cosa obtenim un nombre màxim de nodes de 2^5 =32. Aquest valor es molt inferior als 81, ja que hi ha estats als que amb aquesta representació dels operadors mai si arribarà.

3.- Es tracta d'una cerca a cegues d'un estat objectiu. Aplicant l'algoritme de cerca en amplada troba la solució a l'endevinalla.

3.1.- Quants operadors tindrem? Quins seran aquests operadors? Com relacionen els operadors els estats descrits més amunt?

Tindré 8 operadors:

Operador 1: +(1000)
Operador 2: - (1000)
Operador 3: +(0100)
Operador 4: - (0100)
Operador 5: +(0010)
Operador 6: - (0010)
Operador 7: +(0001)
Operador 8: - (0001)

Cada un dels operadors suma a la tupla de partida la tupla especificada en cada operador.

Per tal d'evitar bucles, un operador només es podrà aplicar si prèviament s'ha aplicat un d'índex i-1 o i-2 i un cop s'ha aplicat no es podrà tornar a aplicar. Dit d'altre manera al nivell i es podran aplicar els operadors i+1 o i+2

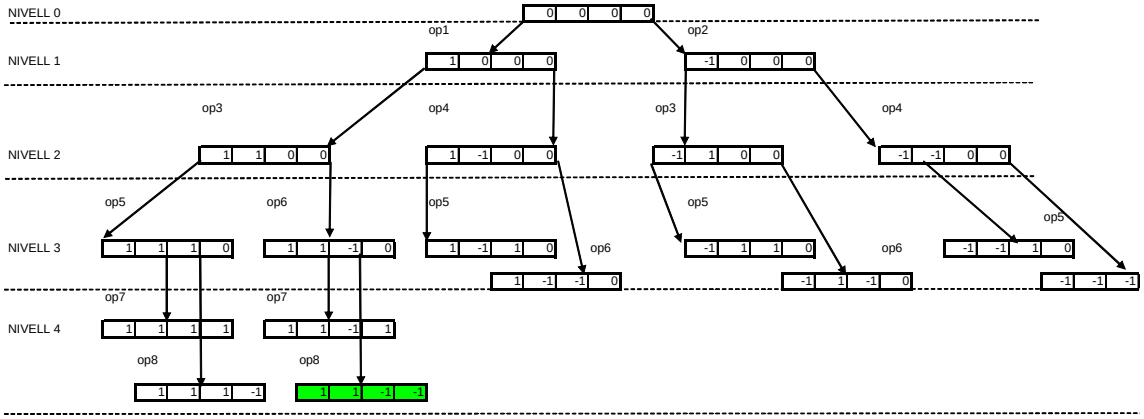
3.2.- Doneu la definició de l'estat inicial i la de l'estat objectiu.
L'estat inicial de la tupla és (0,0,0,0).

L'estat final ve representat implícitament, de forma que s'ha de complir:

$s_i \neq 0$
i

$$\begin{pmatrix}1 & 4 & 3 & 2\end{pmatrix} * \begin{pmatrix}s1\\ s2\\ s3\\ s4\end{pmatrix} = 0$$

3.3.- Feu una representació de l'arbre de cerca.



3.3.a - A quin nivell de profunditat s'ha trobat la solució? 4
3.3.b - Quants nodes heu generat? 19 nodes
3.3.c - Quantitat de memòria necessària (en nombre de nodes).

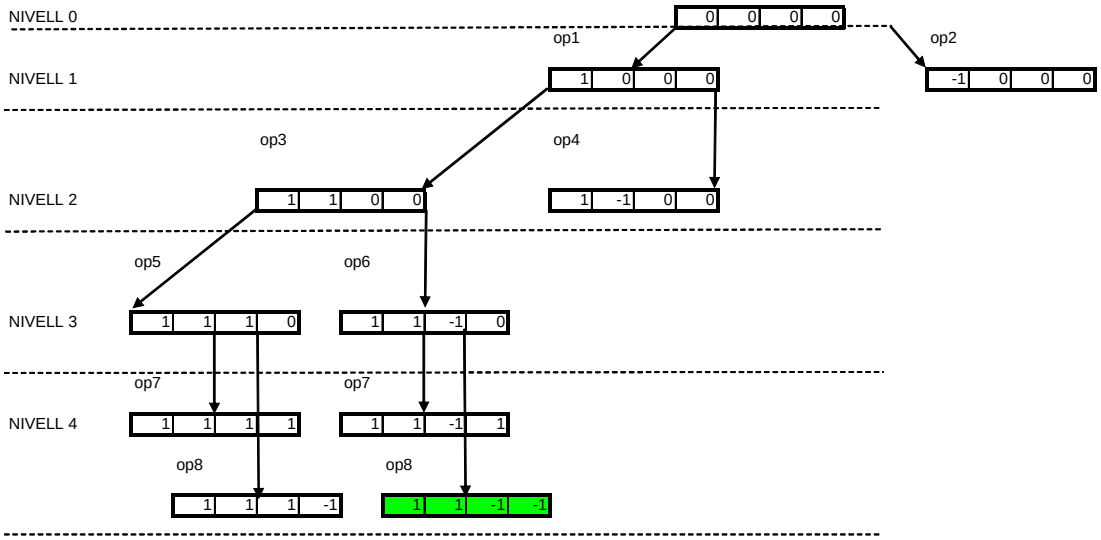
Cua pendents
(0000)
(1000) (-1000)
(-1000) (1100) (1-100)
(1100) (1-100) (-1100) (-1-100)
(1-100) (-1100) (-1-100) (1110) (11-10)
(-1100) (-1-100) (1110) (11-10) (1-110) (1-1-10)
(-1-100) (1110) (11-10) (1-110) (1-1-10) (-1110) (-11-10)
(1110) (11-10) (1-110) (1-1-10) (-1110) (-11-10) (-1-110) (-1-1-10)
(11-10) (1-110) (1-1-10) (-1110) (-11-10) (-1-110) (-1-1-10) (1111) (111-1)
(1-110) (1-1-10) (-1110) (-11-10) (-1-110) (-1-1-10) (1111) (111-1) (11-11) (11-1-1)

La memòria necessària és de 10 nodes, ja que he fet la simplificació d'avaluar en afegir a la cua de pendents.

3.3.d - Nodes pendents d'expansió en trobar la solució.

10 nodes

4.- Ara aplicant l'algoritme de cerca en profunditat troba la solució a l'endevinalla i analitza l'eficiència d'aquest algoritme comparant-la amb la de l'anterior.



Cua pendents
(0000)
(1000) (-1000)
(1100) (1-100) (-1000)
(1110) (11-10) (1-100) (-1000)
(1111) (111-1) (11-10) (1-100) (-1000)
(111-1) (11-10) (1-100) (-1000)
(11-10) (1-100) (-1000)
(11-11) (11-1-1) (1-100) (-1000)

L'algoritme és més eficient ja que totes les solucions tenen el mateix nivell de profunditat i aplicant l'algoritme en profunditat es necessita menys memòria, el nombre màxim de nodes a emmagatzemar és de 5, comparat amb els 10 d'abans.