UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Recursive Best First Search[1]

Alfons Juan
Jorge Civera
Albert Sanchis

DSIC

Departament de Sistemes Informàtics i Computació

[1] Per a una correcta visualització, es requereix Acrobat Reader v. 7.0 o superior.

Objectius

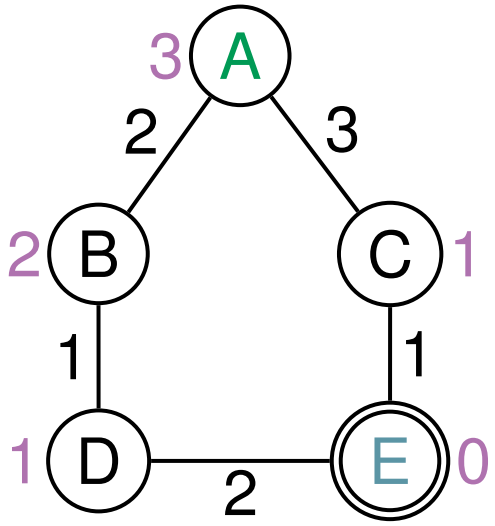
- Aplicar l'algorisme RBFS.
- Construir l'arbre de cerca RBFS.
- Analitzar l'optimalitat i complexitat de la cerca RBFS.

Índex

1 Introducció

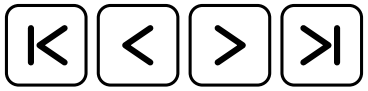
RBFS es basa en cerca amb backtracking delimitada amb un valor f, però, a diferència de IDA*, garanteix explorar primer-el-millor per a funcions d'avaluació monòtones.

Graf d'estats



Cerca

A 0+3	$F_A = 3$	bound = ∞
AB 2+2	$F_B = 4$	
AC 3+1	$F_C = 4$	bound = 4
ABD 3+1	$F_D = 4$	
ABDE 5+0	$F_E = 5$	
ABD	$F_D = 5$	
AB	$F_B = 5$	
AC	$F_C = 4$	bound = 5
ACE 4+0	$F_E = 4$	
ACE sol.!		



RBFS obté el límit del segon-millor valor f de nodes germans en el camí explorat.

2 L'algorisme RBFS (main) [1]

RBFS(G, s', f**)** // G graf ponderat, s' inici, f ***func. aval.***

$P = InitStack(s')$ // Inicialitza ***Path*** amb el node arrel

$b = \infty$ // Límit inicial

$F_{s'} = f_{s'}$ // El valor emmagatzemat és inicialitzat al valor f

$(F_r, r) =$ **BT**$(G, P, F_{s'}, f, b)$ // Torna v. emmagatzemat i est. obj.

if $r \neq$ NULL: **return** P // Si solució, torna ***Path*** a l'objectiu

L'algorisme RBFS (backtracking) [1]

BT(G, P, F_s, f, b) // G graf, P ***Path***, Valor emmagatzemat $F_{s'}$, f, b límit

$s = Top(P)$ // ***Path***: extraure cim de la pila

if $Goal(s)$: **return** (f_s, s) // Solució trobada!

$O = InitQueue()$ // ***Open***: cua de prioritat per a nodes fill

for all $(s, n) \in Adjacents(G, s)$ **and** $n \notin P$: // Generant fills n no en ***Path***

if $f_s < F_s : F_n = max(f_n, F_s)$ // Si s visitat, el fill hereta el v. emmagatzemat

else: $F_n = f_n$ // En altre cas, el v. emmagatzemat és ***f***

$Push(O, n, F_n)$ // Fills ordenats en cua de prioritat per v. emmagatzemats

if $EmptyQueue(O)$: **return** (∞, NULL) // No fills, límit = ∞

while True:

$(n, F_n) = Top(O)$ // Millor fill en funció del valor emmagatzemat F

if $F_n > b$: **return** (F_n, NULL) // S'excedeix el límit, backtracking

$(n', F_{n'}) = Top2(O)$ // 2-millor F o si no existeix, llavors $F_{n'} = \infty$

$Push(P, n)$ // Afegir fill al ***Path*** explorat

$(F_n, r) =$ **BT**$(G, P, F_n, f, min(b, F_{n'}))$ // Recursió amb possible nou límit

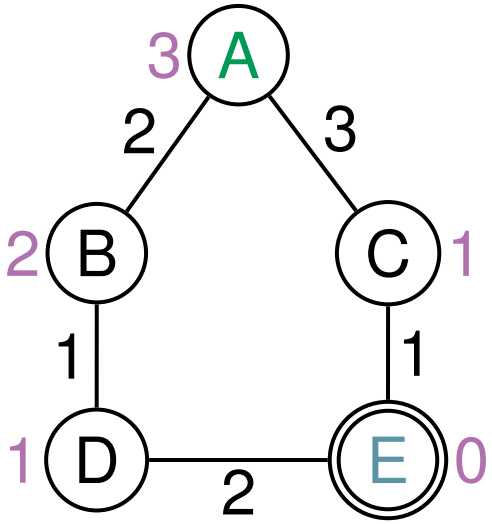
if $r \neq NULL$: **return** (F_n, r) // Si solució, fí recursió sense actualització

$Update(O, n, F_n)$ // Actualitzar node n en O

$Pop(P)$ // Descartar últim fill del ***Path***

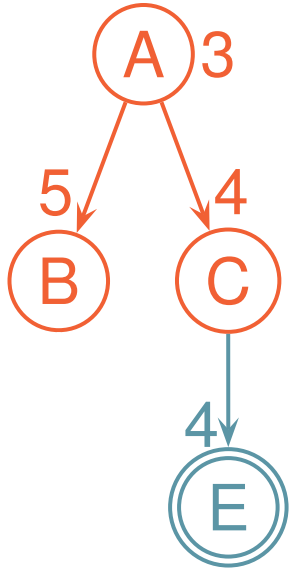
3 Espai de cerca RBFS

Graf d'estats



RBFS

$b = \infty$

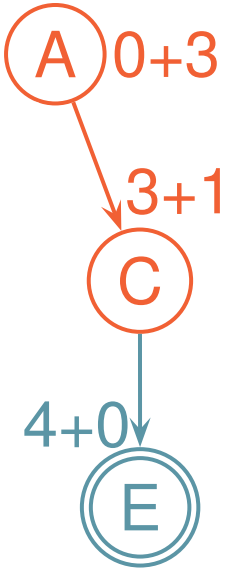


$b = 5$

Path: A3C4B5E4

*IDA**

$b = 4$



$min = 5$

It 2: A3C4E4

4 Propietats

- Un node ha sigut visitat quan $f_s < F_s$, en altre cas $f_s = F_s$
  - $f_s < F_s$: El fill hereta el valor emmagatzemat del pare si $f_n < F_s$
  - $f_s = F_s$: El valor emmagatz. del fill és f_n en la primera exploració
- El límit s'actualitza quan s'entra en la recursió
  - Mínim entre el límit actual y el 2-millor valor emmagatzemat en nodes fills
- F_n es el mínim valor ***f*** del subarbre expandit per baix de n
  - F_n s'actualitza quan s'ix de la recursió
- Nous nodes explorats en ordre primer-el-millor, inclús per a funcions ***f*** no monòtones
- Backtracking sols evita cicles en el ***Path***

5 Optimalitat i complexitat

- ***Completesa:*** Com A* sempre finalitza en grafs finits.
- ***Optimalitat:*** Com primer-el-millor, depèn de la funció d'avaluació.
- ***Complexitat espacial:*** $O(bd)$
- ***Complexitat temporal:*** $O(b^d)$ com IDA*, en la pràctica:
  - Un subconjunt de nodes són re-expandits en cada iteració
  - És necessària la cua de prioritat ***Open*** per als fills de cada node
  - Més eficient en temps que IDA*, re-expansió des del 2-millor

6 Conclusions

Hem estudiat:

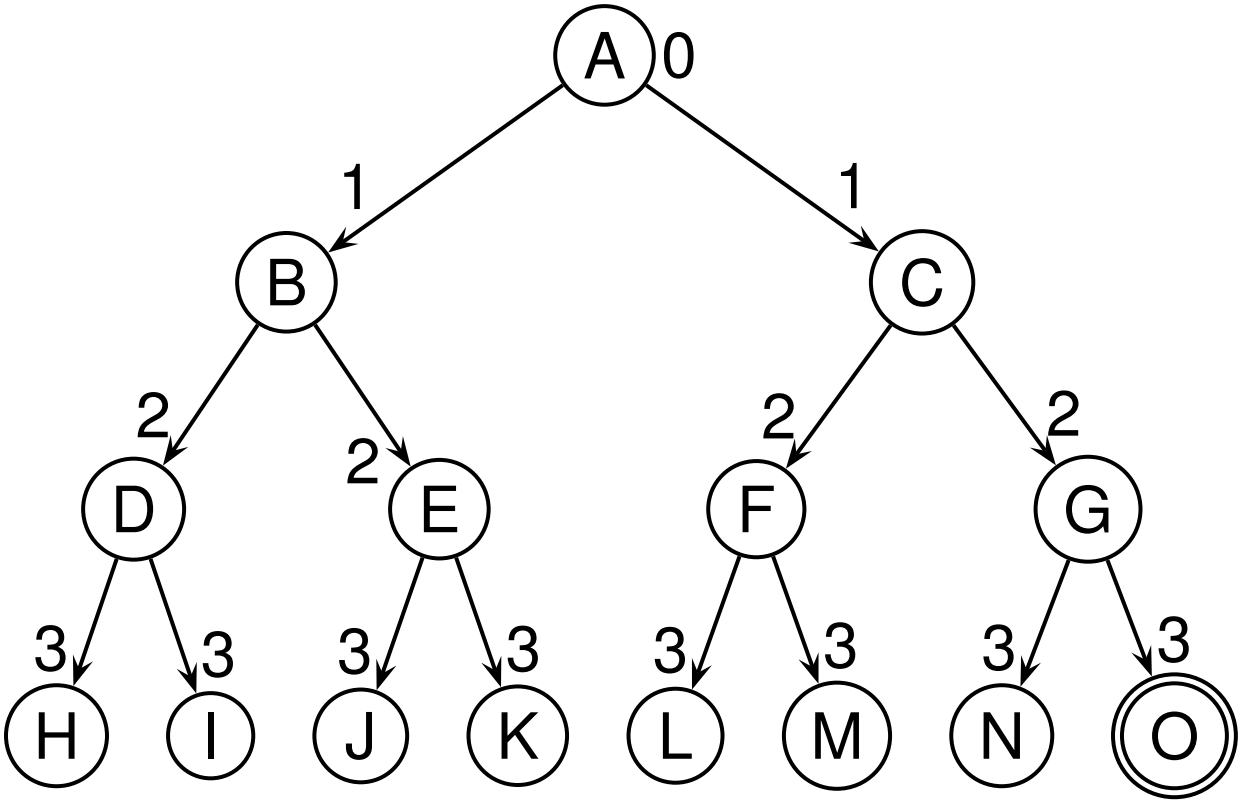
- L'algorisme RBFS.
- L'espai de cerca RBFS.
- Propietats, optimalitat i complexitat de la cerca RBFS.

Alguns aspectes a destacar sobre RBFS:

- Complet i òptim si $f = g + h$ on h es admissible.
- Cost espacial reduït però més que IDA*.
- Cost temporal depèn de la funció d'avaluació ***f***.

Exercici RBFS

Realitza una traça de RBFS en l'espai d'estats de baix (f-valor següent a cada node) i respon a les preguntes del final:



- Màxim nombre de nodes en memòria?
- Nombre total de nodes generats?

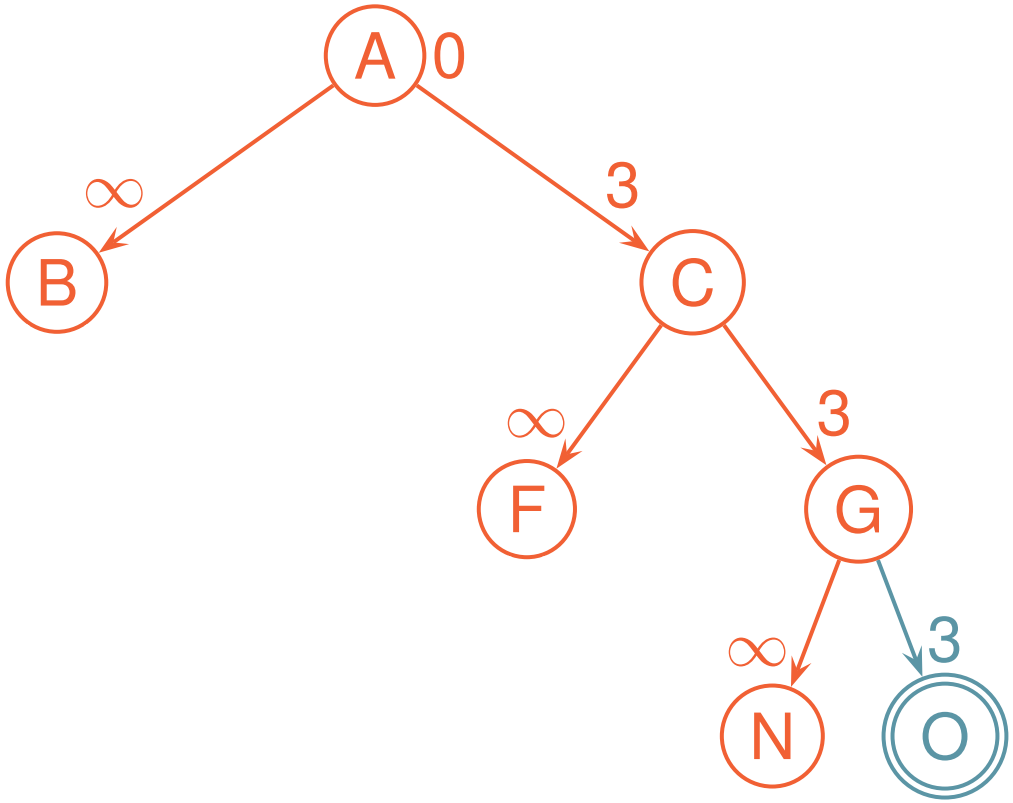
RBFS solució

$b = \infty$

$b = \infty$

$b = \infty$

$b = 3$



Referències

[1] Richard E. Korf. Linear-space best-first search. *Artificial Intelligence*, 62(1):41–78, 1993.