UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Cerca en amplària[1]

Albert Sanchis
Alfons Juan

DSIC

Departament de Sistemes Informàtics i Computació

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

Objectius formatius

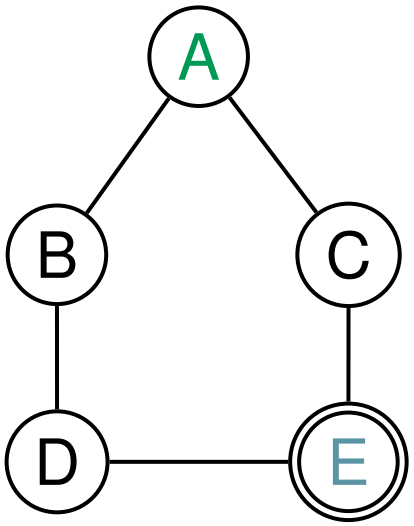
- Descriure cerca en amplària.
- Construir l'arbre de cerca en amplària.
- Analitzar la qualitat i complexitat de cerca en amplària.

Índex

1 Introducció

Cerca en amplària (BFS, de Breadth-first search) consisteix a enumerar camins (des del node inicial) fins a trobar una solució, prioritzant els més curts i evitant cicles (sense repetir nodes):

Graf d'estats



Cerca

A

AB

AC

ABD

ACE sol.!

2 Cerca en amplària [1, 2, 3, 4]

BFS(G, s') // ***Breadth-first search***; G graf i s' node inicial

  $O = IniCua(s')$ // ***Open: frontera-cua*** de la cerca

  $C = \emptyset$ // ***Closed:*** conjunt de nodes explorats

  mentre no $CuaBuida(O)$:

    $s = Desencua(O)$ // selecció ***FIFO (First in, first out)***

    $C = C \cup \{s\}$ // s ja explorat

    per a tota $(s, n) \in Adjacents(G, s)$: // generació: n fill d's

      si $n \notin C \cup O$: // n no descobert fins ara

        si $Objectiu(n)$ **retorna** n // solució trobada!

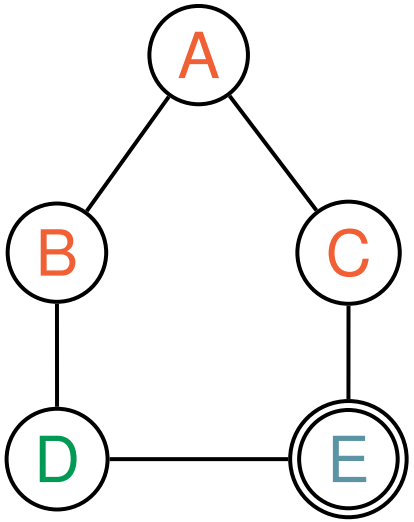
        $Encua(O, n)$ // afegim n a la cua

  retorna NULL // cap solució trobada

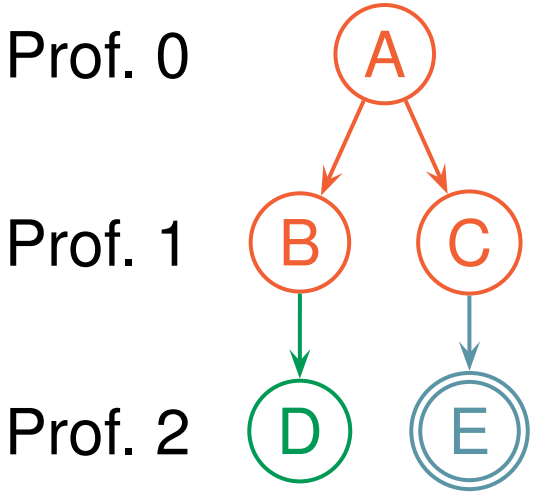
3 L'arbre de cerca en amplària

BFS genera un ***arbre de cerca*** arrelat al node inicial i ***profunditat*** d igual a la llargària "del" (un) camí més curt cap a una solució:

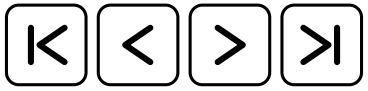
Graf d'estats



Arbre de cerca



It. 3: ABCD E sol.!



Nota: desempatats resolts per ordre alfabètic ("1r l'esquerrà").

4 Completesa, optimalitat i complexitat

- ***Completesa:*** Sí, sempre troba solució (si hi ha una almenys).
- ***Optimalitat:*** Sí, amb accions de cost positiu idèntic.
- ***Complexitat:***
  - $G = (V, E)$ *explícit:* $O(|V| + |E|)$ temporal i espacial.
  - G *implícit,* amb ***factor de brancatge*** b i fins ***profunditat*** d***:***

    Arbre complet (*Pitjor cas*): $O(b^d)$ temporal i espacial.

    Si $Objectiu$ després de selecció: $O(b^{d+1})$ temporal i espacial.

5 Conclusions

Hem vist:

- L'algorisme de cerca en amplària.
- L'arbre de cerca en amplària.
- La qualitat i complexitat de cerca en amplària.

Alguns aspectes a destacar sobre BFS:

- Completa i òptima amb arestes de cost idèntic.
- Cost espacial excessiu, sobretot amb solucions profundes.
- Pot ser una bona opció per a grafs dispersos (poques arestes), solucions superficials i arestes de cost idèntic.

Referències

[1] E. Moore. The shortest path through a maze. In *Proc. of the Int. Symposium on the Theory of Switching, Part II*, pages 285–292. Harvard University Press, 1959.

[2] C. Y. Lee. An algorithm for path connections and its applications. *IRE Trans. on Electronic Computers*, EC-10, 1961.

[3] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, third edition, 2010.

[4] Bernhard Korte and Jens Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, 2018.

bfs.py

```
#!/usr/bin/env python3
from queue import Queue
G={'A':['B','C'],'B':['A','D'],'C':['A','E'],
→ 'D':['B','E'],'E':['C','D']}
def bfs(G,s,t):
→if s==t: return [s]
→O=Queue(); O.put((s,[s])) # Open queue
→OCs=set(); OCs.add(s)     # Open and closed set
→while O:
→→s,path=O.get()
→→for n in G[s]:
→→→if n not in OCs:
→→→→if n==t: return path+[n]
→→→→O.put((n,path+[n]))
→→→→OCs.add(n)
print(bfs(G,'A','E'))
```

bfs.py.out

```
['A', 'C', 'E']
```