Căutare cu revenire

27 aprilie 2004

Metode de proiectare a algoritmilor

Pentru a rezolva o problemă:
– trebuie să alegem ideea potrivită de soluţionare (algoritmul)
– şi structurile de date corespunzătoare pe care operează acesta.

Câteva tehnici generale:
– Căutarea cu revenire (backtracking)
  când soluţia e găsită doar prin încercarea tuturor soluţiile posibile
– Metoda *greedy*
  când soluţia se obţine făcând la fiecare pas mutarea "cea mai bună"
– Descompunerea în subprobleme (divide and conquer)
  prin împărţirea problemei in mai multe probleme similare, mai mici
– Programare dinamică
  tot prin subprobleme mai mici, dar de regulă cu porţiuni comune

Obs. *Recursivitatea* e un element la nivel mai fundamental faţă de metodele de mai sus (toate acestea se pot implementa recursiv)

Căutarea cu revenire

Avem nevoie de:
– o procedură recursivă de căutare
– o structură de date (globală) în care memorăm soluția

procedura caută(s: solutie)
  dacă soluția e bună, tipărește; **return**
  pentru fiecare continuare posibilă
    adaugă continuarea la soluție
    caută(soluție completată)
    șterge continuarea din soluție

Obs: Această variantă ne afișează toate soluțiile.
Pentru a nu afișa decât una, modificăm procedura să returneze dacă o soluție s-a găsit sau nu, și întrerupem ciclul de căutare dacă da.

Exemplu: Colorarea unui graf

Să se coloreze un graf cu un număr dat de culori, cu noduri adiacente colorate diferit.

procedura cauta()
  dacă toate nodurile colorate, afișează; **return**;
  alege un nod necolorat n
  pentru fiecare culoare c
    colorează n cu c
    dacă nu există vecin colorat cu c, cauta()
  decolorează n

Exemplu: Calculul permutărilor

Afișați, în ordine crescătoare, toate permutările numerelor de la 1 la n

Soluție: Pentru fiecare poziție, de la 1 la n, alegem pe rând fiecare din numerele nealese încă, și continuăm cu poziția următoare

procedura alege(p: poziție)
  dacă $p > n$ tipărește permutarea
  pentru i de la 1 la n
    dacă i e liber
      pune i pe poziția p
      marchează i ca ales
      alege(p + 1)
      marchează i ca liber

Exemplu: Găsirea unei căi într-un graf

Într-un graf (ordonat), există un drum de la un nod s la un nod f ?

Soluție: Parcurgem, cu revenire, graful începând de la s, până găsim f sau epuizăm drumurile. Ne oprim din drum când închidem un ciclu.

drum: șir de noduri
procedura caută(n: nod)
  dacă n e f, tipărește drumul curent; **stop**
  pentru fiecare succesor u al lui n
    dacă u nu e pe drumul curent
      adaugă u la drumul curent
      caută(u)
      scoate u din drumul curent

program principal
inițializează drumul cu s
caută(s)