*METODA BACKTRACKING*

*De multe ori, în aplicaţii apar probleme în care se cere găsirea unor soluţii de forma x=x1x2…xn, unde xi∈Ai, i=1,…,n, în care x1...xn trebuie să îndeplinească anumite condiţii. Am putea să generăm toate combinaţiile posibile de valori şi apoi să le alegem doar pe cele convenabile.*

*Considerând mulţimile Ai={ai,1, ai,2,…,ai,n(i)}, aceste combinaţii s-ar putea construi astfel: pentru fiecare valoare posibilă fixată pentru componenta xi, vom alege toate valorile posibile pentru componenta xi+1 şi pentru fiecare astfel de valoare fixată pentru xi+1 vom alege toate valorile posibile pentru componenta xi+2 etc.*

*Rezolvând problema în acest mod, deci generând toate elementele produsului cartezian A1xA2x...An şi verificând abia apoi dacă fiecare combinaţie este o soluţie, eficienţa este scăzută.*

*Astfel, dacă de exemplu ne propunem să generăm toate cuvintele formate cu literele a, b, c, aşa încât fiecare literă să apară o singură dată, combinaţiile posibile sunt în număr de 27, dintre care convin doar 6.*

*Tehnica Backtracking propune generarea soluţiei prin completarea vectorului x în ordinea x1x2...xn şi are la bază un principiu „de bun simţ”: dacă se constată că având o combinaţie parţială de forma v1v2...vk-1 (unde v1, ..., vk-1 sunt valori deja fixate), dacă alegem pentru xk o valoare vk şi combinaţia rezultată nu ne permite să ajungem la o soluţie, se renunţă la această valoare şi se încearcă o alta (dintre cele netestate în această etapă). Într-adevăr, oricum am alege celelalte valori, dacă una nu corespunde nu putem avea o soluţie. […]*

**

*Înainte de a scrie programul care ne va obţine soluţiile, trebuie să stabilim unele detalii cu privire la: vectorul soluţie (câte componente are, ce menţine fiecare componentă), mulţimea de valori posibile pentru fiecare componentă (sunt foarte importante limitele acestei mulţimi), condiţiile de continuare (condiţiile ca o valoare x[k] să fie acceptată), condiţia ca ansamblul de valori generat să fie soluţie.*

*Pe baza acestor date vom scrie apoi procedurile şi funcţiile pe care le vom apela în algoritmul general al metodei […]. Aceste proceduri şi funcţii au o semnificaţie comună, prezentând însă particularităţi în funcţie de fiecare problemă în parte.*

*Astfel, se va nota cu x vectorul care va conţine soluţia; x[k]=v va avea ca semnificaţie faptul că elementul al v-lea din mulţimea de valori posibile Ak a fost selectat pentru componenta xk. Dacă mulţimea Ak are m elemente, a1a2...am, pentru uşurinţă, ne vom referi la ele prin indicii lor, 1, 2, ..., m.*

*Observaţie: De obicei valorile posibile sunt chiar succesive şi în acest caz se poate considera că x[k]=v are semnificaţia că pentru componenta xk s-a ales chiar valoarea v. […]*

*Exerciţii şi probleme […]*

1. *Dacă pentru nivelul k oarecare al vectorului soluţie am verificat toate valorile posibile:*
	1. *algoritmul se încheie;*
	2. *se revine pe nivelul anterior;*
	3. *se trece pe nivelul următor.*
2. *După ce s-a găsit o valoare convenabilă pentru componenta k, următorul pas este:*
3. *se trece la componenta următoare, k+1 (dacă nu s-a ajuns la soluţie);*
4. *se rămâne la componenta k, căutând în continuare o altă valoare convenabilă;*
5. *se revine la componenta k-1.*
6. *În ce condiţii se revine la componenta anterioară?*
7. *după ce am găsit o valoare convenabilă pentru componenta k;*
8. *dacă valoarea testată pentru componenta k nu convine;*
9. *dacă am testat toate valorile posibile pentru componenta k.*

*Fiecărei situaţii din prima coloană îi corespund una sau mai multe operaţii din coloana a doua – realizaţi asocierile corecte.*

|  |  |  |
| --- | --- | --- |
| *Nr.crt* | *Situaţie* | *Operaţie* |
| *1* | *Se revine de la componenta k+1 la k* | *Iniţializare pentru x[k]* |
| *2* | *Se trece de la componenta k-1 la k* | *Se testează următoarea valoare posibilă pentru x[k]* |
| *3* | *Se rămâne pe nivelul k* | *Se tipăreşte o soluţie* |

*(Adaptat după Manualul de Informatică, clasa a X-a, Livia Ţoca, Andreea-Ruxanda Demco, Cristian Opincaru, Adrian Sindile)*