

bile - descrierea soluției

prof. Emanuela Cercez, Liceul de Informatică "Grigore Moisil" Iași

Soluție 1.

Problema cere să generăm toate submulțimile de k elemente ale mulțimii $\{1, 2, \dots, n\}$ astfel încât oricare două submulțimi generate consecutiv să difere printr-un singur element.

Numărul de soluții este, evident, $\text{Comb}(n, k) = n! / (k! \cdot (n-k)!)$.

Algoritmul de generare este inspirat din relația de recurență a lui Pascal:

$$\text{Comb}(n, k) = \text{Comb}(n-1, k) + \text{Comb}(n-1, k-1).$$

Mai exact submulțimile de k elemente ale mulțimii $\{1, 2, \dots, n\}$ pot fi partiționate în două clase: submulțimile care conțin valoarea n (acestea sunt $\text{Comb}(n-1, k-1)$) și submulțimile care nu conțin valoarea n (acestea sunt $\text{Comb}(n-1, k)$).

Să notăm cu $S_{n,k}$ o soluție corectă (adică o succesiunea combinărilor de n luate câte k care respectă condițiile din enunț).

$$S_{n,1} = \{1\} \{2\} \{3\} \dots \{n\}, \text{ pentru orice } n$$

$$S_{n,n} = \{1, 2, 3, \dots, n\}$$

Pentru a genera $S_{n,k}$:

1. vom genera combinările de $n-1$ luate câte k în ordinea specificată (adică $S_{n-1,k}$)
2. vom genera combinările de $n-1$ luate câte $k-1$ în ordinea specificată (adică $S_{n-1,k-1}$), la care reunim elementul n
3. Construim $S_{n,k}$ din $S_{n-1,k}$ urmat de $\underline{S_{n-1,k-1}} \cup \{n\}$ unde cu $\underline{S_{n-1,k-1}}$ am notat $S_{n-1,k-1}$ considerat în ordine inversă.

De exemplu:

$$S_{2,1} = \{1\} \{2\}$$

$$S_{3,1} = \{1\} \{2\} \{3\}$$

$$S_{3,2} = S_{2,2} \quad S_{2,1} \cup \{3\} = \{1, 2\} \{2, 3\} \{1, 3\}$$

$$S_{4,1} = \{1\} \{2\} \{3\} \{4\}$$

$$S_{4,2} = S_{3,2} \quad S_{3,1} \cup \{4\} = \{1, 2\} \{2, 3\} \{1, 3\} \{3, 4\} \{2, 4\} \{1, 4\}$$

$$S_{4,3} = S_{3,3} \quad S_{3,2} \cup \{4\} = \{1, 2, 3\} \{1, 3, 4\} \{2, 3, 4\} \{1, 2, 4\}$$

$$S_{5,2} = S_{4,2} \quad S_{4,1} \cup \{5\} = \{1, 2\} \{2, 3\} \{1, 3\} \{3, 4\} \{2, 4\} \{1, 4\} \{4, 5\} \{3, 5\} \{2, 5\} \{1, 5\}$$

$$S_{5,3} = S_{4,3} \quad S_{4,2} \cup \{5\} =$$

$$\{1, 2, 3\} \{1, 3, 4\} \{2, 3, 4\} \{1, 2, 4\} \{1, 4, 5\} \{2, 4, 5\} \{3, 4, 5\} \{1, 3, 5\} \{2, 3, 5\} \{1, 2, 5\}$$

etc

Se poate demonstra cu ușurință prin inducție că acest procedeu produce o secvență de combinări care respectă condițiile din enunț.

Observați că procedeul de generare este asemănător celui utilizat pentru codul *Gray*.

Vom descrie un algoritm de tip succesori pentru generarea $S_{n,k}$:

Configurația inițială este $s = \{1, 2, \dots, k\}$

Cât timp nu am generat toate combinațiile:

- afișăm configurația curentă;
- generăm configurația următoare, dacă există.

Generarea succesorului:

Pentru a nu trata în mod special cazul ultimului element, $s[k+1] = n+1$;

```
for (j=1; j<=k && s[j]==j; j++);
if (j%2!=k%2)
    if (j==1) s[1]--;
        else {s[j-1]=j; s[j-2]=j-1;}
    else
        if (s[j+1]!=s[j]+1) {s[j-1]=s[j]; s[j]++;}
        else {s[j+1]=s[j]; s[j]=j}
```

Soluție 2 (Zoltan Szabo)

Se cere generarea tuturor combinațiilor a n elemente luate câte k astfel încât două mulțimi succesive să difere printr-un singur element.

Să generăm aceste combinații cu metoda backtracking, folosind o stivă st în care vom pune k elemente în ordine strict crescătoare, generând toate soluțiile în ordine lexicografică.

Putem observa, că în toate cazurile în care ultimul element al stivei are succesori ($st[k] < n$), cerința cerută va fi respectată (fiindcă ultimul elemente cește cu 1, și toate celelalte elemente rămân neschimbate).

În cazurile în care ultimul element al mulțimii nu are succesori ($st[k] = n$), trecerea către următorul element, nu va respecta cerința dorită.

Să observăm, că elementul $st[p]$ nu are succesori în ordine lexicografică, dacă $st[p] - p = n - k$. Deci valorile posibile pentru $st[p]$ sunt incluse în intervalul $[st[p-1], n-p-k]$.

Vom modifica algoritmul anterior, astfel încât elementele stivei să poată să aibă la diferite nivele pașii 1 sau -1 astfel:

- pentru pasul 1 valoarea $st[p] = n-p-k$ nu are succesori,
- pentru pasul -1 valoarea $st[p] = st[p-1]$ nu are predecesori.

Combinări în ordine lexicografică	Pasul corespunzător fiecărui element din stivă	Generar ea bilelor	Pasul corespunzător fiecărui element din stivă	Explicație
1 2 3 4	(1,1,1,1)	1 2 3 4	(1,1,1,1)	Pornim cu prima combinație banală și pt. fiecare nivel pas=1
1 2 3 5	(1,1,1,1)	1 2 3 5	(1,1,1,1)	
1 2 3 6	(1,1,1,1)	1 2 3 6	(1,1,1,1)	
1 2 4 5	(1,1,1,1)	1 2 4 6	(1,1,1,-1)	Când ultimul nu mai are succesori, revenim, generăm succesorul, iar pentru elementele
1 2 4 6	(1,1,1,1)	1 2 4 5	(1,1,1,-1)	
1 2 5 6	(1,1,1,1)			

1 3 4 5	(1,1,1,1)			schimbăm pasul
1 3 4 6	(1,1,1,1)	1 2 5 6	(1,1,1,1)	Penultimul și ultimul nivel nu mai au succesor
1 3 5 6	(1,1,1,1)			
1 4 5 6	(1,1,1,1)	1 3 5 6	(1,1,-1,-1)	La nivelul 2 am trecut la succesor, iar nivelele superioare și-au schimbat pasul
2 3 4 5	(1,1,1,1)	1 3 4 6	(1,1,-1,-1)	
2 3 4 6	(1,1,1,1)	1 3 4 5	(1,1,-1,-1)	
2 3 5 6	(1,1,1,1)	1 4 5 6	(1,1,1,1)	
2 4 5 6	(1,1,1,1)	2 4 5 6	(1,-1,-1,-1)	Etc.etc.
3 4 5 6	(1,1,1,1)	2 3 5 6	(1,-1,-1,-1)	
		2 3 4 6	(1,-1,-1,-1)	
		2 3 4 5	(1,-1,-1,-1)	
		3 4 5 6	(1,1,1,1)	

Să observăm că prin acest procedeu, după ce am generat un element, restul elementelor stivei se pot completa instantaneu până la nivelul k . Astfel avem un algoritm backtracking optimizat, care generează toate soluțiile în ordine corectă fără să treacă prin valori succesive nevalide.

Soluție 3 (80 puncte):

Andrei Grigorean, Universitatea București

Vom construi o procedura recursiva care primind doi parametri N si K va genera toate permutarile de N elemente luate cate K care sa respecte proprietatea din enunt. Vom incerca sa generam mai intai toate combinarile care nu contin elementul N , urmate de toate combinarile care il contin pe N . Pentru a face acest lucru la inceputul procedurii noastre vom face 2 apeluri recursive cu urmatorii parametrii : $N-1$ si K pentru a genera combinarile care nu il contin pe N , urmat de un apel cu parametrii $N-1$ si $K-1$. Pentru combinarile rezultate in urma celui de-al doilea apel, vom adauga la sfarsit elementul N . Astfel am obtinut ceea ce ne-am propus: toate combinarile de N elemente luate cate K . Problema care apare este ca nu avem garantia faptului ca ultima combinatie din primul apel si prima din al doilea respecta conditiile din enunt. Ne vom folosi de urmatoarea observatie:

Avand un sir de combinari de N elemente luate cate K care respecta proprietatea din enunt si o functie bijectiva $f:N \rightarrow N$, inlocuind peste tot un element x cu valoarea $f(x)$, sirul de combinari care rezulta respecta la randul sau proprietatea din enunt.

Consideram ca ultima combinatie rezultata in urma primului apel este $v_1, v_2, \dots, v_{K-1}, v_K$, iar prima combinatie rezultata in urma celui de-al doilea apel este $u_1, u_2, \dots, u_{K-1}$. Ne vom alege o functie $f:(N-1) \rightarrow (N-1)$ astfel incat $f(u_1) = v_1, \dots, f(u_{K-1}) = v_{K-1}$ (pentru elementele x din intervalul $[1 \dots N-1]$ care nu se regasesc printre valorile u_i , $f(x)$ poate lua orice valoare care pastreaza bijectivitatea). Nu este deloc dificil sa gasim o astfel de functie bijectiva. Inlocuind fiecare element x din combinarile rezultate in urma celui de al doilea apel cu $f(x)$, obtinem o solutie valida.