comp - solutie

Algoritmul pe care-l prezint în continuare se bazează pe o analiză atentă a tuturor şirurilor pentru n<12. Scriind un algoritm care foloseşte metoda backtracking, pentru n=3,4,7,8,11,12, dintre toate soluţiile generate reţinem:

Pentru n=3
`2 3 1 2 1 3`
Pentru n=4
`2 3 4 2 1 3 1 4`
Pentru n=7
`2 3 6 2 7 3 4 5 1 6 1 4 7 5`
Pentru n=8
`2 3 6 2 8 3 4 7 5 6 1 4 1 8 5 7`
Pentru n=11
`2 3 6 2 8 3 10 11 5 6 4 9 7 8 5 4 1 10 1 11 7 9`
Pentru n=12
`2 3 6 2 8 3 10 7 12 6 5 11 9 8 4 7 5 10 1 4 1 12 9 11`

Observăm că aceste şiruri luate câte două, pentru valori consecutive ale lui n în mulţimea numerelor naturale de forma 4h sau 4h+3, o secvenţă din primele numere comună. De exemplu pentru n=8 şi n=11 avem secvenţa comună `2 3 6 2`, iar pentru n=11 şi n=12 avem secvenţa comună 2 3 6 2 8 3 10.

Folosind acest lucru putem să preluăm de la două şiruri consecutive secvenţa comună şi să o utilizăm la şirul ce se va genera. Dintre toate aceste şiruri vom alege pe cea care are cu şirul anterior secvenţa comună mai mare.

De exemplu, dacă vrem să generăm un şir pentru `n=12`, folosim secvenţa comună de la şirul pentru `n=8` şi `n=11`, adică `2 3 6 2 8 3` şi obţinem şirul `2 3 6 2 8 3 10 7 12 6 5 11 9 8 4 7 5 10 1 4 1 12 9 11`,
care are cea mai lungă secvenţă comună cu cel de la `n=11`, dintre toate care se pot genera (având ca primă secvenţă 2 3 6 2 8 3), obţinându-se 2 3 6 2 8 3 10.

Prezentăm în continuare acest algoritm.

Structuri de date folosite:
`x` este un tablou unidimensional care va memora sirul ce se va genera
`z` este un tablou unidimensional ce va memora sirul anterior ca dimensiune lui `x`, initial are ca valoare termenii sirului (2,3), adica `(2 3 1 2 1 3)`
`y` este un tablou ce va memora sirul anterior ca dimensiune lui `z`, initial are ca valoare termenii sirului (2,4), adica `(2 3 4 2 1 3 1 4 )`
`n` este numarul pentru care se genereaza sirul, n>4.
`v` este un tablou unidimensional, care are semnificatia: `v[i]=1`, daca perechea `(i,i)` a fost introdusa in `x`, respective `v[i]=0`, in caz contrar.

```
daca not(n mod 4 = 0 or n mod 4 =3) atunci
   scrie ”nu exista sir ”
   * se termina algoritmul
sfarsit daca
pentru m=7,n executa
  nr:=lungimea celei mai lungi secvente comune lui z si y ce incepe cu cifra 2
  y:=z
  max:=0
  pentru i=1,m executa
    v[i]:=0
  sfarsit pentru
  pentru i=1,2*m executa
    x[i]:=0
  sfarsit pentru
  pentru i=1,nr executa
    daca v[y[i]]=0 atunci
```

```
      v[y[i]]:=0
      x[i]:=y[i]
      x[i+1+y[i]]:=y[i]
    sfarsit daca
  sfarsit pentru
```

generam folosind metoda backtracking componentele `x[i], i=1,2*m` *egale cu 0 si dintre acestea oprim in* `z` *pe cel care are secventa comuna cu y, care incepe cu* `2`*,mai mare*

```
sfarsit pentru
pentru i=1,2*n executa
  scrie z[i]
sfarsit pentru
```

Scriind un program in C++ pentru acest algoritm am obtinut (pe un calculator AMD 1800MHz, RAM 256) urmatorii timpi de executie:

```
n=7
2 3 6 2 7 3 4 5 1 6 1 4 7 5
sutimi de sec: 0
n=8
2 3 6 2 8 3 4 7 5 6 1 4 1 8 5 7
sutimi de sec: 0
n=11
2 3 6 2 8 3 10 11 5 6 4 9 7 8 5 4 1 10 1 11 7 9
sutimi de sec: 0
n=12
2 3 6 2 8 3 10 7 12 6 5 11 9 8 4 7 5 10 1 4 1 12 9 11
sutimi de sec: 0
n=15
2 3 6 2 8 3 10 7 12 6 9 15 13 8 14 7 11 10 4 5 9 12 1 4 1 5 13 15 11 14
sutimi de sec: 0
n=16
2 3 6 2 8 3 10 7 12 6 11 15 16 8 13 7 14 10 4 5 9 12 11 4 1 5 1 15 13
16 9 14
sutimi de sec: 0
n=19
2 3 6 2 8 3 10 7 12 6 11 14 17 8 18 7 19 10 13 15 16 12 11 4 9 5 14 1 4
1 17 5 13 18 9 15 19 16
sutimi de sec: 33
n=20
2 3 6 2 8 3 10 7 12 6 11 14 17 8 20 7 15 10 19 16 18 12 11 13 9 5 14 1
4 1 17 5 15 4 9 20 16 13 19 18
sutimi de sec: 55
n=23
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 15 12 11 16 20 18 14
4 13 9 17 5 4 1 19 1 15 5 21 9 16 22 13 23 18 20
sutimi de sec: 132
n=24
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 20 15 13 14
5 18 16 17 4 9 5 19 1 4 1 21 13 15 22 9 23 20 24 16 18
sutimi de sec: 791
n=27
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 27 14
9 18 16 17 20 15 5 19 4 13 9 21 5 4 22 1 23 1 24 16 18 15 25 13 26 20
27
sutimi de sec: 818
n=28
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
27 15 9 17 16 20 18 19 1 13 1 21 9 5 22 4 23 15 24 5 4 16 25 13 26 18
20 28 27
sutimi de sec: 841
n=31
```

```
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
29 30 31 17 5 27 18 19 20 16 5 21 15 9 22 4 23 13 24 1 4 1 25 9 26 18
16 28 15 20 29 13 30 27 31
sutimi de sec: 962
n=32
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
29 30 32 17 31 9 27 19 16 18 20 21 4 15 22 9 23 4 24 13 5 1 25 1 26 16
5 28 18 15 29 20 30 13 27 32 31
sutimi de sec: 1033
n=35
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
29 30 32 17 33 34 35 19 27 31 4 21 16 20 22 4 23 13 24 5 18 15 25 9 26
5 1 28 1 16 29 13 30 9 20 32 27 15 33 18 34 31 35
sutimi de sec: 1505
n=36
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
29 30 32 17 33 34 36 19 35 13 31 21 18 27 22 20 23 16 24 9 4 15 25 13
26 4 1 28 1 9 29 18 30 5 16 32 20 15 33 5 34 27 31 36 35
sutimi de sec: 1994
n=39
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
29 30 32 17 33 34 36 19 37 38 39 21 13 35 22 31 23 27 24 9 16 20 25 15
26 18 13 28 5 9 29 1 30 1 5 32 4 16 33 15 34 4 20 36 18 27 37 31 38 35
39
sutimi de sec: 3982
n=40
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
29 30 32 17 33 34 36 19 37 38 40 21 39 35 22 31 23 27 24 1 16 1 25 20
26 15 9 28 13 4 29 18 30 5 4 32 9 16 33 5 34 15 13 36 20 27 37 31 38 35
18 40 39
sutimi de sec: 5866
n=43
2 3 6 2 8 3 10 7 12 6 11 14 17 8 19 7 21 10 22 23 24 12 11 25 26 28 14
29 30 32 17 33 34 36 19 37 38 41 21 43 40 22 42 23 39 24 35 5 31 25 27
26 16 5 28 18 20 29 13 30 15 9 32 4 1 33 1 34 4 16 36 9 13 37 18 38 15
20 27 41 31 40 35 43 39 42
sutimi de sec: 13968
…
```

Bibliografie

1. On Langford's problem. II., Davies, Roy O., *Math. Gaz.*, 1959, v43, 253-5;
2. Notes on an Algorithm to Enumerate Langford Sequences,© John E. Miller, 1997;
3. Langford's Problem Graphics Gallery, ©1997 by John E. Miller;

Prof. Doru Popescu Anastasiu
Colegiul Naţional ”Radu Greceanu”, Slatina

Solutie Alexandru Mosoi

In literatura aceasta problema poarta numele de "sirul lui Langford". Legenda spune ca Langford s-a inspirat dintr-un joc al copilului sau. A observat ca pentru anumite numere (5, 6, etc) nu exista solutie decat daca se foloseste si numarul 0 o singura data (siruri lacunare).

O solutie deterministica la aceasta problema (utilizand un back destept) nu gaseste o asezare corecta in timp util pentru N > 44 (cea mai buna solutie deterministica a comisiei a obtinut o solutie pentru N = 44 (5 minute)).

O varianta de back ar fi urmatoarea: la fiecare pas cel mai mare numar care nu a fost folosit inca se amplaseaza in fiecare pozitie posibila si se trece la urmatorul pas.

Pentru inceput sa remarcam ca numarul de solutii (pentru N=4k si N=4k+3) este foarte mare si problema cere gasirea unei singure modalitati de asezarea a copiilor. O prima solutie ar fi folosirea unor euristici ca sa mai reduca din cautare (evident nu va gasi anumite solutii).

Rezolvarea comisiei foloseste numere aleatoare pentru determinarea solutiei. Intai plaseaza deterministic copii cu tricorile N, N-1, ..., N-p (p o constanta mica, p=4) iar apoi incearca sa gaseasca in mod aleatoriu o solutie. Pentru gasirea solutiei se modifica back-ul descris mai sus. Mai intai numarul cel mai mare se amplaseaza pe o pozitie aleatoare si apoi pe fiecare pozitie posibila (ca in varianta initiala). Pentru a nu cauta prea mult back-ul se opreste dupa I iteratii (I = 1.000.000) se incearca o noua asezare deterministica a copiilor cu numerele cele mai mari.

Aceasta abordare a gasit solutie pentru N=2.000 in ~200 secude din a doua incercare (prima a esuat).