Soluție problema DIF2

Autori:
prof. Ionel-Vasile Piț-Data, Colegiul Național Traian, Drobeta Turnu Severin
student Lucian Bicsi, Universitatea București

În primul rând, se observa becurile pot fi atinse in orice ordine, fără a produce schimbări de efect în final. O a doua observație utilă în rezolvarea problemei este că, pentru a transforma configurația inițială în cea finala, orice bec ajunge sa fie atins maximum o data. Acest lucru rezultă din faptul că atingerea unui bec pentru a doua oara produce neutralizarea primei operații.

Din cele doua observații se deduce că numărul de seturi de operații diferite posibile este 2^n (orice set de operații se poate vedea ca atingerea unei submulțimi de becuri din cele n).

Soluția pentru 20p:

Pentru 20% din teste, n <= 20, deci se poate simula orice set de operații și, pentru cele care produc efectul dorit, să se calculeze numărul de operații și să se compare cu minimul obținut de până atunci.

Complexitatea acestei soluții este O(2^n).

Soluția pentru 100p:

La o analiză mai atentă a configurațiilor posibile și a modurilor din care se poate ajunge la acestea, se observă că există maximum două soluții de transformare valide. Mai exact, aplicând o strategie tip Greedy pentru a transforma șirul inițial în cel final, se observă că setul de operații este unic determinat de atingerea sau nu a primului element. Mai exact:

Să presupunem că am ales să atingem primul element (cazul în care nu îl atingem se va trata în mod analog). Se disting două cazuri:

1) Becul 1 din starea actuală e diferit de becul 1 în starea finală, unde singura variantă posibilă de a rezolva această problemă este să atingem becul 2 (deoarece becul 1 nu mai poate fi schimbat, din observațiile de mai sus)

2) Stările celor doua becuri coincid, caz în care nu avem voie sa atingem becul 2 (dacă l-am atinge, nu am mai putea modifica înapoi starea becului 1)

Inductiv, se demonstrează că atingerile tuturor celor n becuri sunt unic determinate.
În final, se poate să obținem o stare diferită pentru ultimul bec, caz in care soluția obtinută nu este corectă.

Soluția de 100p tratează cele două cazuri (atingerea sau nu a primului bec), după care simulează strategia descrisă mai sus pentru a obține o configurație care diferă de cea finală la cel mult ultima poziție. Dacă cele doua valori coincid, se actualizează eventualul număr minim de operații. Dacă acest minim nu a fost actualizat, nu vor exista soluții (în cadrul problemei, însă, orice test admite soluție validă).

Extra:
Pentru o mai bună înțelegere a corectitudinii problemei, propunem încă o demonstrație, mai formală:
Datorita comutativității setului de operații, este suficient și necesar să aducem ambele configurații la o stare intermediară.
Să considerăm (S0) = 000...000 și (S1) = 000...001.

Lemă: Din orice stare inițială se poate ajunge în S0 sau S1
Demonstrație: Rezultă din algoritmul descris mai sus (de fiecare dată când găsim un bec aprins, atingem becul imediat următor și repetăm).

În continuare, dacă se poate ajunge din S0 în S1 sau invers, atunci se poate ajunge din orice stare în orice stare. Ne interesează acum să analizăm când se întâmplă acest lucru.

Analizăm în funcție de resturile lui n la împărțirea cu 3:
0) n = 3k: Putem din starea (S1) să atingem becurile 1, 2, 4, 5, 7, 8, ..., 3k-2, 3k-1 și ajungem în (S0)
1) n = 3k + 1: Putem din starea (S1) să atingem becurile 2, 3, 5, 6, ..., 3k - 1, 3k și ajungem în (S0)
2) n = 3k + 2: Acesta este cazul interesant. În acest caz demonstrăm imediat că din (S0) nu se poate ajunge în (S1) (sau invers)

Pentru cazurile în care n = 3k sau n = 3k + 1 rezultă imediat că se poate ajunge din orice stare în oricare alta (deoarece din orice stare se poate ajunge în (S0) sau (S1) și din (S0) se poate ajunge în (S1)).

Cum dintr-o stare x se poate ajunge în toate cele 2^n stări diferite, iar numărul de operații posibile este 2^n, rezultă imediat că modul prin care se ajunge din x în y este unic, pentru orice y (să presupunem că există două submulțimi care îl transformă pe x în y; conform principiului lui Dirichlet, există o stare care nu poate fi atinsă, pentru că există 2^k stări diferite și 2^k transformări diferite, iar două dintre ele duc în aceeași stare => contradicție cu ce am demonstrat mai sus).

În concluzie, pentru n = 3k sau n = 3k + 1, soluția este unică și este obținută prin combinarea (prin ***sau exclusiv***) a celor două șiruri rezultate din algoritmul descris.

Pentru n = 3k + 2, să considerăm starea (S0) și să presupunem prin absurd că putem ajunge în (S1). Atunci, în mod evident, putem ajunge din (S0) în orice altă stare. Folosind un raționament asemănător cu cel din demonstrația precedentă, rezultă că există un singur mod de a transforma (S0) într-o stare x. Ori, există două moduri de a ajunge de la (S0) la (S0): una este setul vid, iar alta este setul de operații 1, 2, 4, 5, 7, 8, ..., 3k + 1, 3k + 2 (se demonstrează prin inducție). Deci presupunerea făcută este falsă. Mai mult, se demonstrează că acest set de operații nevid este unicul mod de a ajunge din (S0) în (S0).

Nota: Același set de operații transformă șirul (S1) în (S1).

De aici se observă că pentru a ajunge dintr-o stare inițială (A) într-o stare finală (B) este suficient și necesar ca:

1) Din stările (A), (B) folosind algoritmul din demonstrația lemei să se ajungă la aceeași stare

2) Setul de operații să fie format din combinarea celor doua seturi de operații din algoritmul respectiv, sau prin combinarea celor două cu "șirul special" 1, 2, 4, 5, ..., 3k + 1, 3k + 2

În concluzie, soluția se poate obține și prin aducerea celor doua șiruri la o formă intermediară (tehnica cunoscută și ca "meet in the middle"), combinarea celor două seturi de operații și, eventual, tratarea cazului special n = 3k + 2 (singurul caz în care există mai multe soluții ale problemei).

Complexitate finală: O(n)