Monede – Mihai Pătraşcu

Problema se poate rezolva în 3 runde în cel mai rău caz, indiferent de N.

Algoritm

În prima rundă comparăm monedele `2k` cu `2k+1`. Fie `S` multimea monedelor `2k` cu proprietatea ca `2k` şi `2k+1` sunt identice. Organizăm `S` ca o listă circulară dublu-înlănţuită (elementele sunt aşezate în listă într-o ordine arbitrară).

La runda `2`, comparăm pe `x` cu `prev(x)` şi `next(x)` – unde `prev` şi `next` se referă la relaţia în lista dublu-înlănţuită. Deoarece lista e circulară, fiecare element din `S` apare în exact două comparaţii. Cum facem asta într-o singură rundă? Păi, ştim ca pentru orice `x` din `S`, `x+1` este egal cu `x`. Deci putem folosi `x` într-o comparaţie şi `x+1` în cealaltă. Folosind rezultatele de la comparaţii, grupăm elementele din listă în grupuri de elemente consecutive care sunt egale; primul element din fiecare grup va fi reprezentantul grupului.

Organizăm reprezentanţii grupului într-o listă circulară dublu-înlănţuită. La a treia rundă comparăm `x` cu `prev(prev(x))` şi `next(next(x))` – unde `prev` şi `next` se referă acum la lista reprezentanţilor. Folosind rezultatele acestor comparaţii, putem afla tipul fiecărui reprezentant. Deci ştim tipul monedelor din fiecare grup, deci putem determina tipul majoritar în mod trivial (prin numărare). Dar cum ştim tipul reprezentanţilor? Primelor două elemente le atribuim arbitrar culorile `1` şi `2`. În continuare examinăm elementele pe rând. Dacă `x` este egal cu `prev(prev(x))`, ştim tipul lui `x` pentru că deja ştim tipul lui `prev(prev(x))`. Dacă `x` este diferit de `prev(prev(x))`, mai ştim şi că el este diferit de `prev(x)` [din modul cum am construit grupurile şi reprezentanţii], aşa că culoarea lui `x` este unic determinată ca fiind singura culoare rămasă.

Demonstrarea optimalităţii

Demonstrăm că nu se poate rezolva problema în mai puţin de 3 runde în cazul cel mai defavorabil, construind un adversar care ne obligă să efectuăm 3 runde.

Orice comparaţii ar efectua algoritmul în prima rundă, adversarul răspunde "egal" la toate. Astfel se crează perechi de elemente care sunt egale pe care le imaginăm "unite" într-un singur element.

La a doua rundă, algoritmul poate face maxim 2 comparaţii cu fiecare element unit (pentru că un astfel de element înseamnă o pereche de elemente). Interpretate ca un graf, comparaţiile cerute formează o reuniune de cicluri şi lanţuri disjuncte. Dacă graful nu e conex (există cel puţin două cicluri/lanţuri), adversarul are o strategie destul de simplă. Pentru orice componentă care conţine mai puţin de jumătate din noduri, adversarul răspunde "egal" la toate comparaţiile. Dacă există o componentă cu mai mult de jumate din noduri (evident aceasta este unică), adversarul răspunde că jumate minus 1 dintre monede sunt de acelaşi tip, şi celelalte sunt de alt tip. Dacă algoritmul încercă să dea răspunsul la problemă, se poate construi uşor un aranjament al monedelor carereprezintă un contrexemplu şi este consistent cu răspunsurile date de adversar. Deci algoritmul trebuie să mai efectueze cel puţin o rundă.

Dacă graful este conex, adversarul răspunde "diferit" la toate comparaţiile. Din nou, dacă algoritmul încearcă să dea răspunsul la problemă, se poate construi uşor un contraexemplu care arată că răspunsul nu e adevărat. Deci algoritmul mai trebuie să efectueze o rundă. În consecinţă, este nevoie de minim 3 runde în cel mai defavorabil caz.