Dist - soluţie

1. Vom extrage cuvintele din prima propoziţie şi le vom reţine într-un vector.
2. Vom elimina spaţiile din cea de a doua propoziţie obţinând astfel un şir de litere (`s`). Problema se reduce la a insera spaţii în acest şir de litere astfel încât `dist(P1,P2)` să fie minimă. Deoarece trebuie să determinăm şi numărul minim de operaţii necesare, vom reţine şi lungimea fiecărui cuvânt din cea de a doua propoziţie (în forma iniţială), precum şi în forma modificată.

Problema se rezolvă prin programare dinamică.

Subproblemă:

Să se determine distanţa minimă dintre propoziţia formată din cuvintele `i`, `i+1`, ... din prima propoziţie şi propoziţia care se poate obţine din caracterele `j`, `j+1`, ... din cea de a doua propoziţie (o vom nota `dist[i][j]`).

`dist[i][j]` este cea mai mică valoare care se poate obţine considerând că vom forma cuvântul curent din cea de a doua propoziţie din `k` litere din `s` ($s_j s_{j+1} \ldots s_{j+k-1}$) ($1 \leqslant k < 20$, cu condiţia să mai existe `k` de litere în `s`), adică este distanţa dintre cuvântul `i` din prima propoziţie şi cuvântul $s_j s_{j+1} \ldots s_{j+k-1}$, la care se adaugă `dist(i+1, j+k)`.

Vom nota cu `nrmin[i][j]`=numărul minim de operaţii necesare pentru a obţine `dist[i][j]`.

`nrmin[i][j]`=este numărul de operaţii necesare pentru a forma al `i`-lea cuvânt din cea de a doua propoziţie din literele `j`, `j+1`, ..., `j+kmin-1` la care se adauga `nrmin[i+1][j+kmin]`, unde `kmin` este acel `k` pentru care se obţine pentru `dist[i][j]` valoarea minimă.

Promo - soluţie

Problema cere determinarea unui graf pe baza grafului muchiilor acestuia (unde 2 muchii sunt adiacente dacă au unul din cele 2 capete în comun). Vom împărţi graful celor M muchii în componente conexe. Pentru fiecare componentă conexă vom ordona muchiile ce fac parte din ea (noduri în grafuri muchiilor) astfel încât orice muchie în afara de prima să aibă cel puţin o muchie adiacentă cu aceasta undeva înaintea ei (folosind o parcurgere DF, de exemplu). Vom determina, pe rând, nodurile ce reprezintă capetele fiecărei muchii din componenta conexă curentă. Primei muchii îi atribuim 2 noduri noi. În continuare, fiecare muchie are cel puţin o altă muchie adiacentă deja poziţionată, astfel că unul din cele 2 capete ale muchiei curente este unul din cele 2 capete ale vecinului ce se află înaintea acesteia în ordonarea pe care am realizat-o. Vom încerca fiecare din aceste 2 capete ca prim capăt al muchiei curente. Drept al doilea capăt vom testa ori un nod neatribuit încă nici unei muchii, ori unul din cele 2 noduri ale unei alte muchii care are capetele deja fixate şi care este vecină cu muchia curentă. O analiză atentă ne va conduce la concluzia că există o singură modalitate de a atribui cele 2 capete muchiei curente în condiţiile în care muchiile dinaintea acesteia au capatele deja fixate. Există doar 2 excepţii posibile :

- când am ajuns la a treia muchie şi aceasta este adiacentă atât cu prima muchie, cât şi cu a doua: cele 3 muchii pot fi aşezate in "stea" sau in "triunghi"
- cand aşezăm a 4-a muchie (sau a 5-a, după caz) poate exista situaţia în care muchiile fixate anterior sunt muchiile unui subgraf bipartit 2x2: în acest caz, muchia curentă poate uni ori cele 2 muchii din partea stângă a subgrafului bipartit, ori cele două noduri din partea dreaptă

Cele 2 excepţii nu pot apărea simultan. În concluzie, pentru fiecare componentă conexă vom avea de încercat cel mult două variante. Complexitatea algoritmului este $O(M^2)$.

Puncte – solutie

Se observa ca pentru fiecare punct `i` exista un interval (posibil nul) pe axa OX in care distanta de la punctele din interval la unul din cele `N` puncte este minima la punctul `i`. Avand calculate aceste intervale pentru fiecare punct, se poate efectua o cautare binara pentru fiecare punct din cele `M` pentru a determina in ce interval apartine, si se afiseaza distanta fata de punctul care reprezinta acel interval.
Pentru a calcula aceste intervale se va mentine o stiva cu intervalele sortate dupa capetele, si punctele care reprezinta acele intervale. La fiecare pas, se introduc punctele in ordinea sortarii, si se determina coordonata `x` la care distanta la punctul curent este mai mica decat distanta la punctul din varful stivei. Daca intervalul aflat in varful stivei are capatul stanga mai mare decat aceasta coordonata `x` determinata, putem elimina acest interval din stiva, deoarece nu va ajuta niciodata la solutie. Acest proces se repeta pana cand se introduc toate punctele, complexitatea acestul pas fiind `O(N)` In final, rezolvarea va avea complexitate `O(N+M*lgN)`.