Pembahasan
Penyisihan Competitive Programming
Tingkat Mahasiswa

CompFest 2013

Kontributor:

- **Alham Fikri Aji**
- **Cakra Wishnu Wardhana**
- **Kemal Maulana Kurniawan**
- **William Gozali**

Panggilan Akrab

Alham Fikri Aji

Tingkat Kesulitan

Mudah

Kategori

Ad hoc, *string matching*, *brute force*

Solusi

Soal bonus untuk penyisihan mahsiswa. Yang perlu dilakukan benar-benar dilakukan apa adanya: untuk setiap pasang orang, periksa apakah terdapat panggilan akrab yang ambigu untuk mereka. Jika ya, tambahkan jawaban dengan satu. Kompleksitas solusi $O(N^2L^2)$ cukup untuk *accepted*.

Mainan Baru

Cakra Wishnu Wardhana

Tingkat Kesulitan

Mudah

Kategori

Ad hoc, *brute force*

Solusi

Bila benar-benar dicoba semua kemungkinan mengoperasikan K giliran itu, ada 2^K langkah. Tentunya kompleksitas sampai 2^K terlalu lambat untuk *accepted*.

Bila diteliti lebih lanjut, operasi geser kiri yang dilanjutkan geser kanan tidak merubah konfigurasi barisan. Semua kemungkinan hasil akhir yang dicapai sebenarnya setara dengan K kali geser kiri, K-2 kali geser kiri, K-4 kali geser kiri, ..., K-4 kali geser kanan, K-2 kali geser kanan, dan K kali geser kanan. Sehingga hanya ada K/2 kemungkinan. Di antara semua kemungkinan itu, coba satu per satu dan ambil yang paling kecil secara leksikografis.

Membandingkan dua barisan secara leksikografis dapat dilakukan dalam O(N). Karena ada K/2 kemungkinan barisan, maka kompleksitas akhirnya adalah O(NK).

Berti Sang Galau

Kemal Maulana Kurniawan

Tingkat Kesulitan

Sedang

Kategori

Matematika, kombinatorik, *divide and conquer*

Solusi

Perhatikan bahwa setiap bit dari N faktor keunikan itu independen terhadap bit lainnya dalam menentukan bit yang bersesuaian pada X. Jadi, cukup dihitung berapa banyak konfigurasi biner yang jika semuanya di-XOR menghasilkan masing-masing bit pada x. Misalkan x_i adalah bit ke-i pada X. Jika $x_i = 0$ maka banyaknya bit 1 pada posisi ke-i dari ke-N bilangan tersebut haruslah genap. Sebaliknya, jika $x_i = 1$ maka banyaknya bit 1 pada posisi ke-i dari ke-N bilangan tersebut haruslah ganjil.

Pada kasus yang genap, banyaknya konfigurasi bit yang mungkin adalah :
$${}_NC_0 + {}_NC_2 + {}_NC_4 + ... + {}_NC_{N-1}$$

Sedangkan pada kasus yang ganjil, banyaknya konfigurasi bit yang mungkin adalah :
$${}_NC_1 + {}_NC_3 + {}_NC_5 + ... + {}_NC_N$$

Sekarang, perhatikan bahwa :
$${}_NC_0 = {}_NC_N$$
$${}_NC_1 = {}_NC_{N-1}$$
$${}_NC_2 = {}_NC_{N-2}$$
...
$${}_NC_N = {}_NC_0$$

Sehingga sebenarnya:
$${}_NC_0 + {}_NC_2 + {}_NC_4 + ... + {}_NC_{N-1} = {}_NC_1 + {}_NC_3 + {}_NC_5 + ... + {}_NC_N$$

Sekarang perhatikan bahwa:
$${}_NC_0 + {}_NC_2 + {}_NC_4 + ... + {}_NC_{N-1} + {}_NC_1 + {}_NC_3 + {}_NC_5 + ... + {}_NC_N = {}_NC_0 + {}_NC_1 + {}_NC_2 + ... + {}_NC_N = 2^N$$

Oleh karena itu, ${}_NC_0 + {}_NC_2 + {}_NC_4 + ... + {}_NC_{N-1}$ sama dengan $2^N/2$, atau 2^{N-1}. Demikian pula untuk ${}_NC_0 + {}_NC_2 + {}_NC_4 + ... + {}_NC_{N-1}$.

Artinya, baik x_i 1 maupun 0, banyaknya kemungkinan ke-N bit itu supaya kalau di-XOR-kan semua hasilnya sama dengan x_i adalah 2^{N-1}. Karena terdapat K bit, maka total semua kemungkinan caranya adalah $2^{(N-1)K}$.

Bagaimana cara menghitung $2^{(N-1)K}$ secara efisien? Anda dapat menggunakan pemangkatan cepat dengan prinsip *divide and conquer* sehingga kompleksitasnya menjadi O(log NK).

Jual Kebun

William Gozali

Tingkat Kesulitan

Sedang

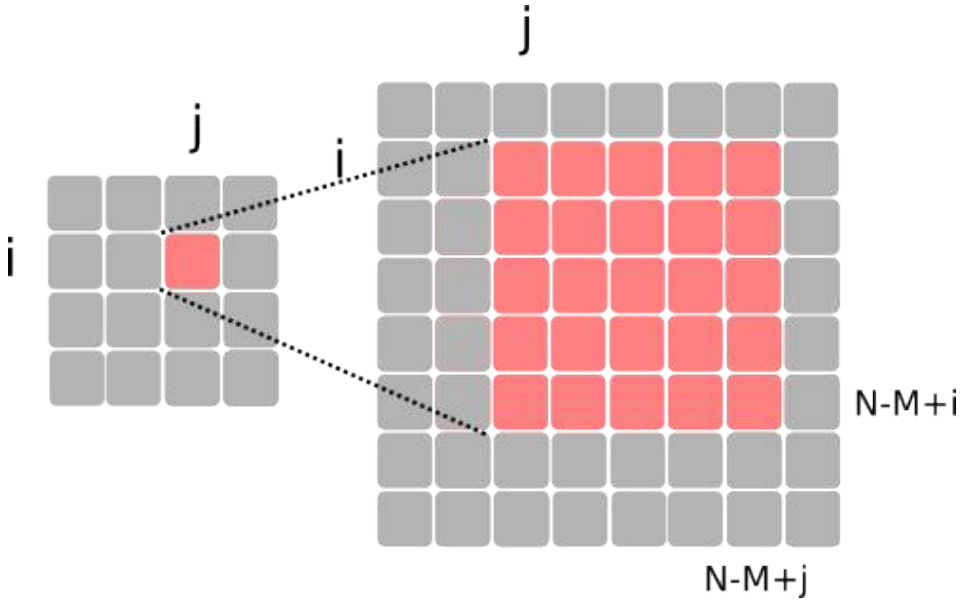
Kategori

prekomputasi, DP *partial sum*

Solusi

Cara naif memerlukan kompleksitas $O(N^2M^2)$, dimana untuk $(N-M+1)^2$ kemungkinan pembelian kebun kita cek satu per satu petaknya dalam $O(M^2)$. Berhubung N dan M cukup besar, cara ini tidak mendapatkan *accepted*.

Untuk mengoptimisasinya, kita perlu melihat persoalan ini bukan dari keseluruhannya. Perhatikan suatu petak yang ingin dibeli oleh pelanggan itu, misalnya petak di baris i dan kolom j. Di mana saja petak ini bisa ditempatkan pada kebun besar itu? Jawabannya adalah di petak a kolom b, yang memenuhi $i \le a \le N-M+i$ dan $j \le b \le N-M+j$. Perhatikan gambar berikut:



Petak di baris i kolom j hanya mungkin di tempatkan di daerah yang beri warna merah (sebelah kanan). Kita tinggal menghitung di sana ada berapa petak yang isinya berbeda dengan petak baris i kolom j yang diinginkan pembeli. Untuk itu, kita bisa menggunakan DP *partial sum*. Anggap ubi sebagai angka 1 dan singkong angka 0. Bila petak baris i kolom j yang diinginkan

pembeli adalah 1, tinggal cari berapa banyak angka 0 di daerah tersebut. Demikian pula sebaliknya.

Dengan begitu, kompleksitas solusi adalah $O(N^2)$ untuk membuat tabel *partial sum*, ditambah perhitungan petak salah yang bisa dilakukan dalam $O(M^2)$. Kompleksitas akhirnya $O(N^2 + M^2)$.

Angkutan Kota

Alham Fikri Aji

Tingkat Kesulitan

Sedang

Kategori

Graph, shortest path, Dijkstra

Solusi

Terdapat dua komponen biaya dalam persoalan ini, yaitu tingkat polusi udara dan waktu. Observasi yang sangat penting adalah, karena fungsi biaya ($P = Ax + By$) itu linier, maka perhitungannya dapat dipecah-pecah. Jadi, apabila perjalanan dilakukan dari persimpangan 1 ke persimpangan 2 dengan biaya (w, x), dilanjutkan dengan perjalanan dari persimpangan 2 ke persimpangan tujuan dengan biaya (y, z), maka biayanya dapat dihitung dengan $P = (Aw + Bx) + (Ay + Bz)$. Hal ini juga berlaku apabila perjalanan yang dilakukan lebih dari 2 tahap.

Dengan begitu, dua komponen biaya tinggal disatukan untuk setiap *edge*, lalu persoalan ini tereduksi menjadi persoalan *shortest path* yang klasik. Anda dapat menggunakan algoritma seperti Dijkstra atau Floyd Warshall (karena banyaknya *node* hanya sedikit).

Kompleksitas solusi bergantung dengan algoritma yang Anda gunakan. Dengan Dijkstra, kompleksitasnya $O(E \log (V + E))$ atau $O(V^2)$, sementara dengan Floyd Warshall didapat $O(V^3)$. Keduanya cukup untuk mendapatkan *accepted*.