Divide and Conquer Lanjutan

Tim Olimpiade Komputer Indonesia

Pendahuluan

Melalui dokumen ini, kalian akan:

- Memahami konsep meet in the middle
- Memahami konsep perpangkatan matriks

Mulai dari bab ini, seluruh kode program akan dituliskan dalam bahasa pseudo-C++.

Meet in the Middle

- Teknik **Meet in the Middle** digunakan dengan membagi masalah menjadi dua submasalah.
- Pencarian lengkap akan dilakukan untuk setiap submasalah dan memeriksa apakah terdapat irisan pada kedua pencarian lengkap tersebut.
- Contoh soal: Diberikan suatu himpunan S berisi N bilangan. Tentukan apakah terdapat subhimpunan dari S yang berjumlah 0.
- Tentunya terdapat solusi menggunakan pencarian penuh dalam waktu $O(2^N)$, namun kita akan menggunakan solusi yang lebih cepat.

Meet in the Middle (lanj.)

- Soal ini dapat diselesaikan dengan membagi himpunan S menjadi dua subhimpunan A dan B sedemikian sehingga $|A| = \lfloor \frac{N}{2} \rfloor$ dan $|B| = \lceil \frac{N}{2} \rceil$.
- Terdapat subhimpunan dari S yang berjumlah 0 jika dan hanya jika terdapat subhimpunan $A' \subseteq A$ dan subhimpunan $B' \subseteq B$ sedemikian sehingga jumlah seluruh bilangan pada A' adalah negasi dari jumlah seluruh bilangan pada B'.
- Untuk setiap subhimpunan $B' \subseteq B$, kita dapat menyimpan jumlah seluruh bilangannya dalam *array* $P(B)$ dan mengurutkannya.
- Untuk setiap subhimpunan $A' \subseteq A$, kita dapat memeriksa apakah negasi dari jumlah seluruh bilangannnya terdapat pada *array* $P(B)$ menggunakan *binary search*.
- Kompleksitas dari solusi ini adalah $O(2^{\lfloor \frac{N}{2} \rfloor} \times \log(2^{\lceil \frac{N}{2} \rceil}))$.

Perpangkatan Matriks

- Teknik perpangkatan matriks dapat menghitung nilai dari matriks M^K untuk sebuah matriks M berukuran $N \times N$ dalam waktu $O(N^3 \times \log(K))$.
- Teknik ini sebenarnya serupa dengan perpangkatan bilangan bulat.

$$M^x = \begin{cases} M, & \text{jika } x = 1 \\ (M^{\frac{x}{2}})^2, & \text{jika } x \text{ bernilai genap} \\ (M^{\lfloor \frac{x}{2} \rfloor})^2 \times M, & \text{jika tidak} \end{cases}$$

Perpangkatan Matriks (lanj.)

Kode di bawah ini dapat digunakan untuk menghitung perpangkatan tipe data apapun.

```
template<typename T>
T exp(T M, int K) {
  if (K == 1) {
    return M;
  }
  T res = exp(M, K >> 1);
  res = res * res;
  if (K & 1) {
    res = res * M;
  }
  return res;
}
```

Perpangkatan Matriks (lanj.)

Untuk menghitung perpangkatan matriks menggunakan kode pada halaman sebelumnya, maka kita dapat mendefinisikan tipe data matriks dan fungsi perkalian.

```
template<typename T> struct Matrix {
  vector<vector<T>> M;

  Matrix operator*(const Matrix& other) {
    Matrix res = (Matrix){vector<vector<T>>(
        M.size(), vector<T>(other.M[0].size(), 0))};

    for (int i = 0; i < M.size(); ++i) {
      for (int j = 0; j < other.M[0].size(); ++j) {
        for (int k = 0; k < M[0].size(); ++k) {
          res.M[i][j] += M[i][k] * other.M[k][j];
        }
      }
    }

    return res;
  }
};
```

Perpangkatan Matriks: Fibonaci

- Salah satu penggunaan perpangkatan matriks adalah untuk menghitung f_N (bilangan fibonaci ke-N) dalam $O(\log(N))$.
- Untuk seluruh bilangan bulat positif n, perhatikan bahwa $f_{n+2} = f_n + f_{n+1}$, sehingga

$$\begin{bmatrix} f_n & f_{n+1} \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} f_{n+1} & f_{n+2} \end{bmatrix}$$

- Sehingga, untuk setiap bilangan bulat positif k

$$\begin{bmatrix} f_n & f_{n+1} \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^k = \begin{bmatrix} f_{n+k} & f_{n+k} \end{bmatrix}$$

Perpangkatan Matriks: Fibonaci (lanj.)

- , Sehingga, nilai f_N dapat diperoleh dengan menghitung nilai

$$\begin{bmatrix} f_1 & f_2 \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^{N-1} = \begin{bmatrix} f_N & f_{N+1} \end{bmatrix}$$

- Nilai $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^{N-1}$ dapat dihitung menggunakan perpangkatan matriks dalam $O(\log N)$.