Nama : Muhammad Faturrahman

NIM : 3312501046

Kelas : IF 1B Pagi

Tugas Kombinatorial Bagian 2

Soal:

1. Sebuah sistem cloud memiliki 5 server, 20 file, dan setiap file disimpan di 2 server untuk redundansi.
   a.Tentukan banyak kombinasi penyimpanan yang mungkin.
   b. Analisis kemungkinan tabrakan data (collision).
   c. Buat simulasi logika / pseudocode deteksi duplikasi.
   Jawab:
   a. Untuk 1 file, banyak kombinasi server yang mungkin adalah:

$$C(n,k) = \binom{n}{k} = \frac{n!}{k!\,(n-k)!}$$

   Di mana $n = 5$ (jumlah server) dan $k = 2$ (jumlah server per file).

$$C(5,2) = \frac{5!}{2!\,(5-2)!} = \frac{5!}{2!\,3!} = \frac{5x4}{2x1} = 10$$

   Ada 10 cara untuk menyimpan satu file.
   Karena ada 20 file yang disimpan secara independen, dan setiap file memiliki 10 kemungkinan lokasi, maka total banyak kombinasi penyimpanan yang mungkin untuk seluruh sistem adalah:

$$\text{Total Kombinasi} = (\text{kombinasi perfile})^{\text{jumlah file}}$$

$$\text{Total Kombinasi} = 10^{20}$$

   b. Jumlah Slot Penyimpanan: Ada 10 pasangan server unik ($C(5,2) = 10$).
   Jumlah Item: Ada 20 file yang harus disimpan (masing-masing menempati 1 pasangan server/slot).
   Karena jumlah file (20) lebih besar dari jumlah total pasangan server unik (10), maka pasti akan terjadi tabrakan. Ini adalah aplikasi langsung dari Prinsip Sarang Merpati (Pigeonhole Principle).
   Pigeonholes (Sarang Merpati): 10 pasangan server unik.
   Pigeons (Merpati): 20 file.
   ika kita menempatkan 20 merpati ke dalam 10 sarang, setidaknya satu sarang akan berisi $[\frac{20}{10}] = 2$ merpati. Artinya, setidaknya satu pasangan server akan menyimpan dua file berbeda.

   c.
```
FUNCTION DeteksiDuplikasiPenyimpanan(DaftarFile):
    PenyimpananMap = New Map<String, Daftar<ID_File>>

    FOR EACH File IN DaftarFile:
        Server1 = File.LokasiServer[0]
        Server2 = File.LokasiServer[1]
```

```
            IF Server1 > Server2 THEN
                SWAP(Server1, Server2)
            END IF

            PasanganKey = KONKATENASI(Server1, ",", Server2)
            FileID = File.ID


            IF PenyimpananMap.HAS_KEY(PasanganKey) THEN
                PenyimpananMap.GET(PasanganKey).ADD(FileID)


                CETAK("Tabrakan Terdeteksi:")
                CETAK("Pasangan Server:", PasanganKey)
                CETAK("File yang Terlibat:",
    PenyimpananMap.GET(PasanganKey).Daftar()
            ELSE
                PenyimpananMap.PUT(PasanganKey, New
    Daftar<ID_File>())
                PenyimpananMap.GET(PasanganKey).ADD(FileID)
            END IF

        END FOR

        CETAK("--- Laporan Akhir ---")
        TabrakanDitemukan = FALSE
        FOR EACH Key, DaftarFileDiMap IN PenyimpananMap:
            IF DaftarFileDiMap.UKURAN() > 1 THEN
                CETAK("Pasangan Server", Key, "menyimpan",
    DaftarFileDiMap.UKURAN(), "file.")
                TabrakanDitemukan = TRUE
            END IF
        END FOR

        IF NOT TabrakanDitemukan THEN
            CETAK("Tidak ada tabrakan penyimpanan antar file.")
        END IF

    END FUNCTION
```

2. Apa keterkaitan antara konsep kombinasi dan desain system penyimpanan data?
   Jawab:
   1. Pemilihan replika = kombinasi/pasangan: memilih subset server untuk menyimpan setiap file adalah masalah kombinatorik (memilih k dari m). Jumlah kombinasi memengaruhi distribusi data dan kemungkinan kemacetan/kesamaan penempatan.
   2. Fault tolerance & redundancy: kombinasi replika menentukan toleransi terhadap kegagalan: jika replika disebarkan pada kombinasi server yang “independen”, maka risiko kehilangan data lebih kecil.

3. Load balancing: desain harus memilih kombinasi penempatan yang merata (karena banyak kombinasi secara acak menghasilkan variasi beban); kombinatorika membantu merancang strategi deterministik/terstruktur (round-robin, consistent hashing, combinatorial block design) untuk mengurangi hot-spots.
4. Combinatorial designs (mis. block designs, Steiner systems) dipakai untuk optimasi replikasi sehingga setiap pasangan/kelompok server berinteraksi dengan pola tertentu berguna untuk diagnosa, recovery paralel, dan pengurangan overlap.
5. Ruang keadaan (state space): jumlah konfigurasi (kombinasi) menentukan kompleksitas manajemen, testing, dan verifikasi sistem.
6. Keamanan & privasi: beberapa kombinasi penempatan dapat mengekspos data ke lebih sedikit perangkat → pertimbangan kombinatorik untuk minimalkan risiko.

3. Mengapa prinsip sarang merpati menjadi penting dalam pemrograman dan struktur data?
   Jawab:
   Prinsip Sarang Merpati adalah alat matematika yang menjamin adanya duplikasi atau tabrakan ketika jumlah item (merpati) melebihi jumlah wadah (sarang). Dalam ilmu komputer, ini krusial untuk:
   1. Struktur Data (Hashing)
      PSP menjamin terjadinya tabrakan (collision) pada hash table.
   2. Batasan Algoritma (Kompleksitas)
      PSP digunakan untuk menetapkan batasan bawah (lower bound) teoritis pada kinerja algoritma.
   3. Kompresi dan Redundansi Data
      PSP mendefinisikan batas kemampuan sistem penyimpanan data.

   - Kompresi: Menjamin bahwa kompresi lossless (tanpa kehilangan data) tidak dapat mengecilkan setiap input unik, karena jika itu mungkin, kita akan memiliki lebih banyak input (merpati) daripada output yang unik (sarang).
   - Redundansi: Dalam sistem terdistribusi, PSP digunakan untuk memastikan adanya duplikasi data yang memadai untuk mencapai toleransi kesalahan (fault tolerance), karena menempatkan lebih banyak salinan (merpati) daripada lokasi unik (sarang) menjamin data disimpan di banyak tempat.
     Singkatnya, PSP adalah prinsip yang membantu kita memahami kepastian tabrakan dan batasan teoretis dalam merancang sistem komputasi yang efisien dan andal.