Menghitung Permutasi Suatu Barisan Menggunakan Kode Lehmer dan Faktoradik

Okaswara Perkasa (13510051)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
okaswara@students.itb.ac.id

***Abstrak* — Makalah ini membahas mengenai algoritma perhitungan permutasi suatu barisan dengan menggunakan teori kode lehmer dan faktoradik. Algoritma yang dibahas adalah mencari permutasi ke-n suatu barisan, serta mencari nilai permutasi ke-n jika diketahui permutasinya. Sebelumnya, makalah ini akan membahas teori mengenai barisan dan permutasinya, kode lehmer, faktoradik, serta algoritma-algoritma dasar pada kode lehmer dan faktoradik.**

***Kata kunci* — algoritma, faktoradik, kode lehmer, permutasi**

I. PENDAHULUAN

Tanpa menggunakan komputer, perhitungan permutasi suatu barisan sebenarnya dapat dilakukan dengan mudah, selama jumlah elemen tidak terlalu besar. Ini dapat dilakukan dengan menuliskan setiap permutasi secara berurutan pada secarik kertas, lalu menghitung banyaknya permutasi sampai pada permutasi yang diinginkan, baik dengan menggunakan perhitungan kombinasional, ataupun dengan menghitung jumlah elemen secara manual.

Namun, seiring dengan bertambahnya jumlah elemen, perhitungan menjadi lebih sulit dan membosankan, serta tingkat kesalahan juga menjadi semakin besar. Perhitungan untuk jumlah elemen yang besar tentu lebih baik jika dilakukan oleh komputer.

Makalah ini akan membahas mengenai salah satu cara perhitungan permutasi suatu barisan dengan menggunakan kode lehmer dan faktoradik. Algoritma perhitungan ini nantinya dapat diaplikasikan pada komputer, ataupun sebagai metode alternatif pada saat melakukan perhitungan secara manual.

Berikut adalah dua permasalahan utama yang akan dibahas dan dipecahkan dalam makalah ini:

1. Diketahui suatu barisan berhingga S yang elemen-elemennya unik. Bagaimana cara mencari permutasi ke-n dari barisan tersebut?
2. Diketahui suatu barisan berhingga S yang elemen-elemennya unik, dan salah satu permutasinya S'. Permutasi ke berapakah S'?

II. DASAR TEORI

A. Barisan (Sequence)

Barisan adalah urutan objek. Hampir sama seperti himpunan, namun urutan elemen yang berbeda berarti barisan yang berbeda. Barisan dinyatakan dalam notasi $(a_0, a_1, ..., a_n)$.

Contoh 2.1 Himpunan $\{A, B, C\} = \{B, C, A\}$, namun barisan $(A, B, C) \neq (B, C, A)$.

Setiap posisi pada barisan dapat diberikan suatu indeks. Pada makalah ini, pemberian indeks adalah nol dari kiri.

Contoh 2.2 Pada barisan (A, B, C), elemen pertama barisan (A) memiliki indeks 0, sedangkan elemen kedua (B) berindeks 1, dan elemen ketiga (C) berindeks 2.

B. Permutasi pada Barisan

Permutasi dari suatu barisan S adalah barisan baru yang memiliki anggota yang sama dengan barisan asli (barisan S), namun urutannya dapat berbeda.

Contoh 2.3 Misalkan terdapat suatu barisan $S = (A, B, C)$, maka barisan (B, C, A) dan (C, A, B) merupakan permutasi dari barisan S.

Jika seluruh kemungkinan permutasi barisan S (termasuk barisan S itu sendiri) dituliskan, lalu diurutkan menaik, maka akan didapatkan suatu barisan S_p yang berisi seluruh permutasi yang terurut menaik secara alfabetis (*lexicographic*). Pada makalah ini, barisan ini diberi nama "barisan permutasi"

Contoh 2.4 Untuk barisan S, barisan permutasinya adalah: S_P = ((A, B, C), (A, C, B), (B, A, C), (B, C, A), (C, A, B), (C, B, A)). Perhatikan bahwa permutasi-permutasi ini dituliskan terurut secara alfabetis, seperti urutan pada suatu kamus.

Banyaknya anggota barisan permutasi adalah faktorial dari banyaknya anggota barisan yang asli, atau $|S_p| = |S|!$. Ini didapatkan dari fakta bahwa barisan permutasi berisi seluruh kemungkinan permutasi dari barisan asli. Maka, anggotanya pastilah sama dengan banyaknya permutasi barisan yang asli.

Pada suatu barisan permutasi S_p, kita dapat menetapkan urutan permutasi, tergantung dari urutannya pada barisan S_p dihitung dari sebelah kiri. Anggota paling kiri dianggap sebagai permutasi ke-0, atau memiliki nilai yang sama dengan indeks barisan S_p.

Contoh 2.5 Berikut beberapa urutan permutasi dari barisan S.

(A,B,C) ← Permutasi ke-0, indeks 0

(A,C,B) ← Permutasi pertama, indeks 1

(B,A,C) ← Permutasi kedua, indeks 2

(B,C,A) ← Permutasi ketiga, indeks 3

dst.

C. Kode Lehmer

Kode lehmer adalah suatu kode yang digunakan untuk merepresentasikan suatu permutasi dari barisan S. Nama kode ini diambil dari Derrick H. Lehmer, seorang matematikawan Amerika.

Kode ini dinyatakan dalam suatu barisan berbentuk $(a_{n-1},\ldots,a_1,a_0)$, dengan n merupakan banyaknya anggota barisan S. Nilai a_{n-1} s.d. a_0 merupakan suatu bilangan bulat, yang menyimpan informasi mengenai urutan anggota barisan S pada permutasi yang ingin direpresentasikan. Atau dapat dikatakan bahwa setiap permutasi dari suatu barisan dapat dituliskan dalam kode lehmer.

Berikut algoritma untuk menghitung kode lehmer untuk suatu barisan S dengan salah satu permutasinya S'.

Algoritma mengubah permutasi menjadi kode lehmer

1. Ambil elemen pertama dari S'.
2. Hitung berapa banyak elemen pada sebelah kanan yang "lebih besar" (dalam arti, pada S elemen tersebut memiliki indeks lebih kecil) dari S' pada elemen tersebut.
3. Tuliskan nilai tersebut sebagai kode lehmer pada indeks nol.
4. Ulangi langkah 1-3 untuk elemen-elemen S' selanjutnya, dan juga kode lehmer selanjutnya. Lakukan pengulangan hingga semua elemen S' telah diproses.

Contoh 2.6 Misalkan kita ingin merepresentasikan salah satu permutasi $S=(A,B,C,D)$, yakni $S'=(C,B,D,A)$ dalam kode lehmer. Elemen yang diberi tanda pada merupakan elemen yang dipilih pada langkah algoritma di atas.

Langkah	S'	Kode Lehmer
1	C, B, D, A	(?,?,?,?)
2	C, B, D, A	(?,?,?,?)
3	C, B, D, A	(2,?,?,?)
1	C, B, D, A	(2,?,?,?)
2	C, B, D, A	(2,?,?,?)
3	C, B, D, A	(2,1,?,?)
1	C, B, D, A	(2,1,?,?)
2	C, B, D, A	(2,1,?,?)
3	C, B, D, A	(2,1,1,?)
1	C, B, D, A	(2,1,1,?)
2	C, B, D, A (tidak ada)	(2,1,1,?)
3	C, B, D, A	(2,1,1,0)

Tabel 2.1

Maka, permutasi (C,B,D,A) dapat dinyatakan sebagai $(2,1,1,0)$ dalam kode lehmer.

Untuk lebih memahami secara intuitif arti barisan yang direpresentasikan oleh kode lehmer, kita akan mencoba melakukan analisis, dengan contoh data yang sama seperti Contoh 2.6.

Berikut potongan barisan permutasi sampai dengan permutasi (C,A,B,D).

(A,B,C,D)
(A,B,D,C)
...
(B,A,C,D)
...
(C,A,B,D)

Ilustrasi 2.1

Kita sudah mengetahui bahwa $S'=(C,B,D,A)$ dan kode lehmer-nya $(2,1,1,0)$. Nilai "2" pada elemen pertama kode lehmer menunjukkan banyaknya "perubahan" yang dilakukan pada elemen pertama dari permutasi, sebelum mencapai elemen "C". Ini sesuai dengan barisan permutasi, bahwa sebelum mencapai "C", elemen pertama haruslah "melewati" dua elemen terlebih dahulu, yakni "A" dan "B".

(C,A,B,D)
...
(C,B,A,D)

Ilustrasi 2.2

Penalaran yang serupa juga dilakukan pada elemen kedua. Elemen "A" harus dilewati terlebih dahulu sebelum mencapai "B". Maka dari itu, nilai elemen kedua kode lehmer adalah satu.

(C,B,A,D)
(C,B,D,A)

Ilustrasi 2.3

Sama seperti sebelumnya, elemen "A" dilewati terlebih dahulu sebelum elemen "D".

Harap diperhatikan bahwa elemen-elemen yang telah digunakan sebelumnya (pada contoh elemen "C" dan "B") tidak muncul pada kedua permutasi pada Ilustrasi 2.3. Ini dikarenakan keduanya telah diletakkan pada

elemen pertama dan kedua. Oleh karena itu, seolah-olah permasalahan dapat diubah seperti mencari kode lehmer untuk permutasi (D, A) dari barisan (A, D). (Hasilnya adalah $(1,0)$, sesuai dengan kedua elemen terakhir kode lehmer $(2,1,1,0)$).

Karena nilai kode lehmer menunjukkan jumlah perubahan elemen, maka terdapat nilai maksimum untuk suatu kode lehmer.

Misal terdapat kode lehmer dalam bentuk $(a_{n-1},...,a_1,a_0)$, yang terdiri dari $n-1+1=n$ elemen. Maka untuk suatu a_x, terdapat maksimum x elemen yang dapat mendahului elemen yang ditujukan. Maka dari itu, kode lehmer maksimum untuk suatu barisan dengan jumlah elemen n adalah $(n-1, n-2,...,1,0)$. Kode ini merepresentasikan permutasi terakhir dari suatu barisan.

Contoh 2.7 Untuk n = 4, kode lehmer yang paling besar adalah $(3,2,1,0)$.

Dengan pemahaman mengenai nilai yang ditunjukkan oleh kode lehmer, kita dapat dengan mudah merancang algoritma yang mengubah suatu kode lehmer menjadi permutasi. Barisan yang dipakai adalah barisan S dengan kode lehmer L. Algoritma ini akan menghasilkan permutasi S'.

Algoritma mengubah kode lehmer menjadi permutasi

1. Ambil elemen pertama dari kode lehmer L, misalkan nilainya x.
2. Ambil elemen ke-(x+1) dari barisan S, dengan tidak memperhitungkan elemen yang pernah diambil sebelumnya/belum terdapat di S'.
3. Tuliskan elemen tersebut sebagai elemen pertama dari S'.
4. Ulangi langkah 1-3 untuk elemen kode lehmer L selanjutnya serta elemen S' selanjutnya. Lakukan pengulangan hingga seluruh elemen L diproses.

Contoh 2.8 Misalkan kita ingin mencari permutasi dari $S=(A, B, C, D)$ yang memiliki kode lehmer $L=(3,1,1,0)$. Elemen yang digarisbawahi dan berwarna merah adalah elemen yang di-"ambil" pada langkah 2. Sedangkan elemen yang dicoret adalah elemen yang sudah terpakai/ada di S'.

Langkah	x	S	S'
1	3	A, B, C, D	
2	3	A, B, C, D	
3	3	A, B, C, ~~D~~	D
1	1	A, B, C, ~~D~~	D
2	1	A, B, C, ~~D~~	D
3	1	A, ~~B~~, C, ~~D~~	D, B
1	1	A, ~~B~~, C, ~~D~~	D, B
2	1	A, ~~B~~, C, ~~D~~	D, B
3	1	A, ~~B~~, ~~C~~, ~~D~~	D, B, C
1	0	A, ~~B~~, ~~C~~, ~~D~~	D, B, C
2	0	A, ~~B~~, ~~C~~, ~~D~~	D, B, C
3	0	~~A~~, ~~B~~, ~~C~~, ~~D~~	D, B, C, A

Tabel 2.2

D. Faktoradik

Faktoradik adalah suatu sistem basis angka yang memakai nilai faktorial sebagai basisnya. Tidak seperti basis pada umumnya (biner, octal, desimal, dan heksadesimal), basis pada faktoradik tidak konstan, namun berubah untuk setiap sukunya.

Berikut representasi sembarang bilangan bulat positif m pada faktoradik, untuk banyaknya digit basis faktoradik sebesar n.

$$(m)_{10} = a_{n-1}(n-1)!+a_{n-2}(n-2)!+...+a_1(1)!+a_0(0)!$$
$$= (a_{n-1}a_{n-2}..a_1a_0)_!$$

Agar terdapat korespondensi satu-satu antara nilai sebenarnya (berbasis 10) dengan faktoradik, nilai a_x harus berada pada selang $0 \le a_x \le x$ untuk x suatu bilangan bulat positif.

Contoh 2.9 Contohnya jika m = 52, maka jika m dinyatakan dalam faktoradik:

$$(52)_{10} = 2\cdot 4!+1\cdot 3!+0\cdot 2!+1\cdot 1!+0\cdot 0!$$
$$= (21010)_!$$

Sebenarnya istilah "faktoradik" masih termasuk baru dalam dunia matematika. Sifat basis faktorial yang dimiliki faktoradik sudah ditemukan sebelumnya, mungkin sudah beberapa kali oleh orang yang berbeda-beda. Namun dokumentasi mengenai basis ini tidak begitu banyak.

Dari berbagai sumber, istilah "faktoradik" kemungkinan besar dipopulerkan pertama kali oleh Dr. James McCaffrey, melalui artikelnya di MSDN yang berjudul "*Using Permutations in .NET for Improved Systems Security"* yang dibuat pada tahun 2003.

Sekilas mengenai faktoradik. Lalu dimana guna faktoradik dalam perhitungan permutasi? Faktoradik dapat digunakan untuk langsung mendapatkan kode lehmer yang sesuai, jika kita ingin mengetahui permutasi ke-n dari suatu barisan.

Pengubahannya sangat mudah untuk dilakukan. Kita cukup mengubah n menjadi basis faktoradik, lalu dari nilai faktoradik $(a_{n-1}a_{n-2}..a_1a_0)_!$, kita dapat langsung mendapatkan kode lehmer yang sesuai, yakni $(a_{n-1}, a_{n-2},..a_1, a_0)$. Ya, kita cukup mengubah setiap digit dari faktoradik menjadi elemen pada kode lehmer.

Contoh 2.10 Contohnya $m=(52)_{10}=(21010)_!$, maka kode lehmer untuk permutasi ke-52 pada suatu barisan yang terdiri dari 5 elemen adalah $(2,1,0,1,0)$.

Lalu bagaimanakah algoritma yang baku untuk mengubah suatu angka menjadi faktoradik? Algoritma untuk melakukan hal tersebut dapat dilihat di bawah. Algoritma yang dipakai sekilas hampir sama dengan algoritma euclid untuk mencari FPB dua buah bilangan.

Untuk lebih memudahkan nantinya, algoritma ini langsung menyimpan nilai-nilai faktoradik pada elemen-elemen kode lehmer.

Misal m adalah elemen yang akan diubah menjadi faktoradik, dan nilai faktoradik tersebut langsung disimpan pada kode lehmer L.

Algoritma mencari kode lehmer yang merepresentasikan permutasi ke-n

1. Tuliskan kode lehmer terakhir (a_0) dengan nilai 0.
2. Bagi m dengan n = 2. Misalkan hasilnya adalah q dan sisa pembagiannya r. Nilai r merupakan nilai kode lehmer kedua sebelum terakhir (a_1)
3. Ganti nilai m dengan q dan tambah nilai pembagi n dengan 1.
4. Ulangi langkah 2-3 hingga nilai m = 0. Gunakan indeks kode lehmer selanjutnya.

Contoh 2.11 Misalkan kita ingin mencari kode lehmer untuk permutasi ke-11. Maka, jika kita menggunakan algoritma di atas:

$$m = q \cdot n + r$$
$$11 = 5 \cdot 2 + 1$$
$$5 = 1 \cdot 3 + 2$$
$$1 = 0 \cdot 4 + 1$$

Kode lehmer untuk permutasi ke-11 adalah $(1,2,1,0)$.

Algoritma kebalikan, yakni mencari nilai n dari suatu kode lehmer, tidak sulit untuk dibuat. Kita cukup mengalikan masing-masing elemen dengan nilai faktorial, sesuai dengan representasi faktoradik. Namun algoritma baku yang dapat diterapkan pada komputer tetap diperlukan.

Algoritma mencari nilai permutasi ke-n yang direpresentasikan dengan kode lehmer.

1. i = 1.
2. Kalikan a_i dengan i!, lalu tambahkan pada nilai n.
3. Tambah nilai i dengan 1.
4. Ulangi langkah 2-3 sampai seluruh elemen kode lehmer diproses.

Contoh 2.12 Menggunakan Contoh 2.12, misal kita ingin mengetahui nilai permutasi ke-n yang direpresentasikan oleh kode lehmer $(1,2,1,0)$. Maka nilai n:

$$1 \cdot 3! + 2 \cdot 2! + 1 \cdot 1! = n = (11)_{10}$$

Elemen kode lehmer yang paling kanan, a_0, tidak ikut diperhitungkan pada algoritma ini. Ini dikarenakan nilai a_0 selalu bernilai nol, sehingga tidak akan memengaruhi hasil nilai n.

II. ALGORITMA UNTUK MEMECAHKAN PERMASALAHAN UTAMA

Setelah membahas dasar teori serta sejumlah algoritma dasar, maka selanjutnya kita akan membahas algoritma yang digunakan untuk menyelesaikan dua permasalahan utama.

Sebenarnya, dengan menggunakan teori serta algoritma-algoritma yang telah dibahas sebelumnya, penyelesaian kedua masalah ini sudah menjadi lebih mudah. Kita tinggal menggabungkan algoritma-algoritma tersebut.

A.Algoritma menghitung permutasi ke-N dari suatu barisan

Seperti yang sudah dibahas, algoritma yang dipakai hanyalah penggabungan dua buah algoritma pada Dasar Teori. Secara garis besar, algoritmanya adalah sebagai berikut, untuk barisan S dan nilai N.

1. Ubah nilai N dalam faktoradik, lalu ubah menjadi kode lehmer.
2. Cari permutasi dari S dengan menggunakan kode lehmer yang didapat.

Kedua langkah diatas adalah dua buah algoritma yang sudah dibahas pada Dasar Teori. Berikut contoh prosedur algoritma ini dalam *pseudocode*.

```
procedure NtoPermutation (input Sn-1,...,S0 : character,
                          input N : integer,
                          output Pn-1,...P0 : character)
{ Menghitung permutasi ke-N dari barisan Sn-1,...,S0 }

Deklarasi
    i, j, k, q, cnt, Nt : integer
    Ln-1,...,L0 : integer { kode lehmer }
    Un-1,...,U0 : boolean { penanda apakah elemen Si sudah
                            diambil}

Algoritma

    { Ubah N menjadi kode lehmer }
    L0 ← 0
    i ← 1
    Nt ← N
    while Nt != 0 do
        q <= floor(Nt/(i+1))
        Li <= Nt mod (i+1)
        Nt <= q
    endwhile
    { Ln-1,...,L0 sudah berisi kode lehmer untuk N }

    { Cari permutasi yang direpresentasikan oleh L }
    { Inisialisasi U }
    i ← 0
    while i < n do
        Ui ← false
    endwhile
    j ← n-1
    { Untuk setiap elemen pada L }
    while j >= 0 do
        k ← n-1
        cnt ← 0
        { Cari indeks elemen S yang belum diambil }
        while cnt <= Lj do
            if not Uk then
                cnt ← cnt+1
            endif
            k ← k-1
        endwhile
        k ← k+1
        { Indeks elemen S sudah didapatkan }
        Pj ← Sk
        Uk ← true { Elemen sudah dipakai }
        j ← j-1 { Proses elemen L selanjutnya }
    endwhile

    { Permutasi P berhasil didapatkan }
```

Kode 2.1

Contoh algoritma di atas memakai array untuk menyimpan data barisan, permutasi, serta kode lehmernya. Terdapat suatu array dari boolean yang menandakan apakah elemen pada S sudah dipakai sebelumnya.

Perhatikan bahwa terdapat variabel “n” dan “N”. Keduanya menunjukkan dua hal yang berbeda. “n” berarti banyaknya elemen pada S, sedangkan “N” adalah nilai permutasi ke-N.

B.Algoritma menghitung nilai N dari suatu permutasi

Sama seperti yang pertama, untuk permasalahan kedua, secara garis besar algoritmanya adalah sebagai berikut:

1. Cari kode lehmer dari permutasi P
2. Ubah kode lehmer kedalam faktoradik, lalu ubah faktoradik kedalam desimal.

Kedua algoritma diatas juga sudah dibahas sebelumnya, jadi kita tinggal menggabungkan kedua algoritmanya saja. Algoritma menghitung nilai N dalam *pseudocode*:

```
procedure PermutationToN (input Sn-1,...,S0 : character,
                          input Pn-1,...P0 : character,
                          output N : integer)
{ Menghitung nilai permutasi dari P untuk barisan Sn-1,...,S0 }

Deklarasi
    i, j, k, q, cnt, Nt : integer
    Ln-1,...,L0 : integer { kode lehmer }
    Un-1,...,U0 : boolean { penanda apakah elemen Si sudah
                            diambil}

Algoritma

    { Inisialisasi U }
    i ← 0
    while i < n do
        Ui ← false
    endwhile
    { Hitung kode Lehmer untuk P}
    j ← n-1
    { Untuk setiap elemen pada P }
    while j >= 0 do
        k ← n-1
        cnt ← 0
        { Hitung berapa banyak elemen lain sebelumnya }
        while Sk ≠ Pj then
            if not Uk then
                cnt ← cnt+1
            endif
            k ← k-1
        endwhile
        { Banyaknya elemen lain sudah didapatkan }
        Lj ← cnt { Lakukan assignment pada elemen L }
        Uk ← true { Elemen sudah dipakai }
        j ← j-1 { Proses elemen P selanjutnya }
    endwhile
    { Ln-1,...,L0 sudah berisi kode lehmer P }

    { Ubah kode lehmer menjadi urutan permutasi (N) }
    i ← 1
    N ← 0
    while i < N do
        N ← N+Li*(i!)
    endwhile

    { Nilai N berhasil dihitung }
```

Kode 2.2

Contoh algoritma Kode 2.2 diatas juga memakai array untuk menyimpan data-data yang diperlukan.

KESIMPULAN

Pencarian permutasi suatu deret memang dapat dengan mudah dilakukan dengan menggunakan kode lehmer dan faktoradik.

Kode lehmer digunakan sebagai "perantara" antar representasi dari permutasi. Sedangkan faktoradik digunakan untuk mencari suatu kode lehmer yang menunjukkan suatu urutan permutasi tertentu.

Penggunaan kode lehmer dan faktoradik pada pencarian permutasi dapat diproses oleh komputer, dengan cara dijadikan algoritma dalam suatu bahasa pemrograman. Algoritma yang dihasilkan juga ringkas dan mudah dipahami.

REFERENSI

[1] http://en.wikipedia.org/wiki/Factorial_number_system. Tanggal akses : 10 Desember 2011.

[2] http://en.wikipedia.org/wiki/Permutation. Tanggal akses : 10 Desember 2011.

[3] http://lin-ear-th-inking.blogspot.com/2012/11/enumerating-permutations-using.html. Tanggal akses : 10 Desember 2011.

[4] http://www.keithschwarz.com/interesting/code/?dir=factoradic-permutation. Tanggal akses : 10 Desember 2011.

[5] http://en.wikipedia.org/wiki/Sequence. Tanggal akses : 11 Desember 2011.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Desember 2011

Okaswara Perkasa (13510051)