Algoritma Decrease and Conquer

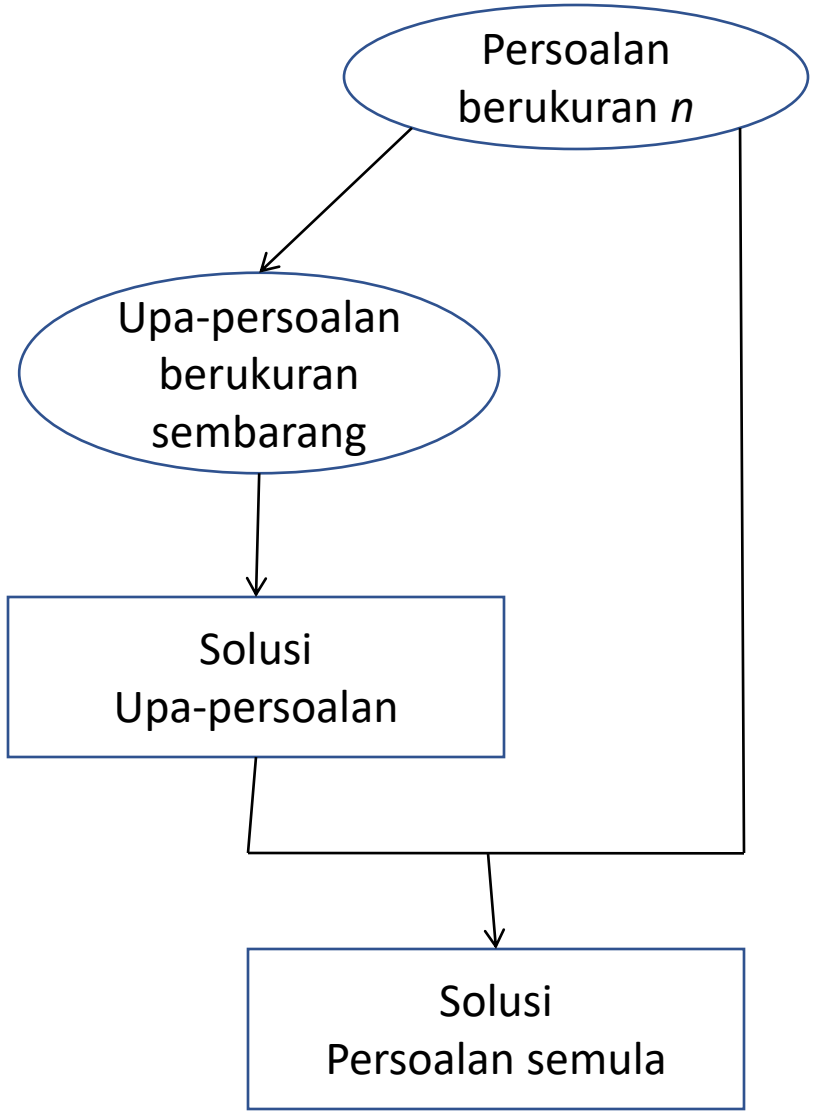
(Bagian 2)

Bahan Kuliah IF2211 Strategi Algoritma

Oleh: Rinaldi Munir

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika ITB
2025

Decrease by a Variable Size



Contoh persoalan:

1. Interpolation search
2. Mencari nilai median

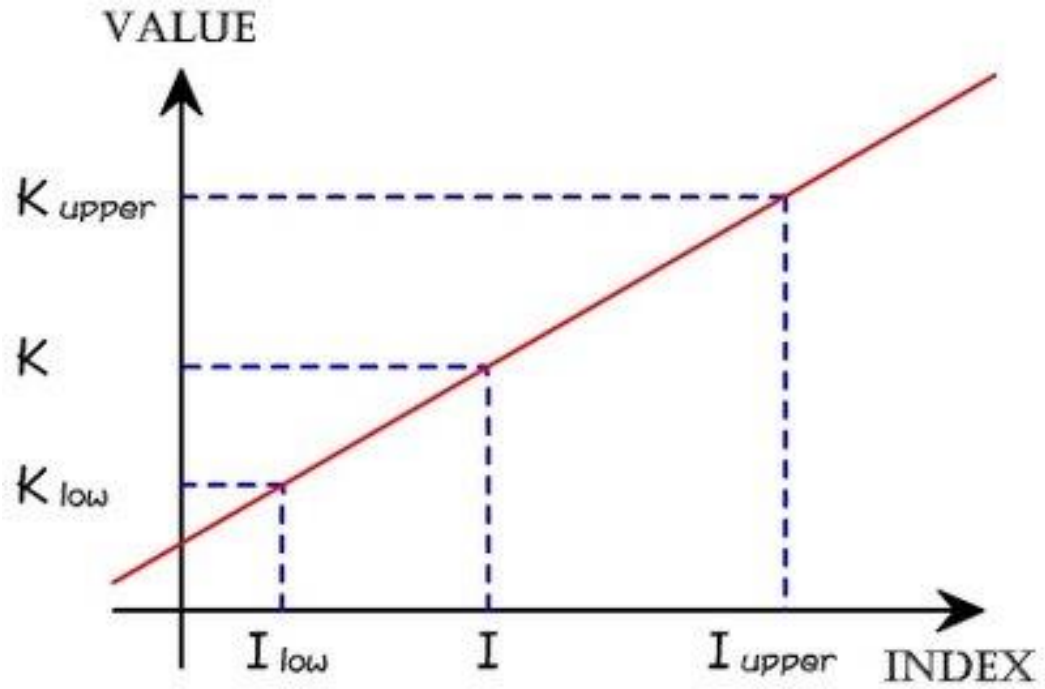
7. *Interpolation Search*

- Algoritma pencarian ini mirip dengan pencarian kata di dalam kamus atau di dalam ensiklopedi dengan cara memperkiraan letak kata tersebut di di dalam kamus

- Semua entri di dalam kamus sudah terurut menaik (dari A sampai Z).

- Memperkirakan letak kata di dalam larik dilakukan dengan teknik interpolasi.

- Kondisi awal: - larik A sudah terurut menaik
  - *K* adalah nilai yang dicari



Perbandingan:

$$\frac{K - K_{low}}{K_{upper} - K_{low}} = \frac{I - I_{low}}{I_{upper} - I_{low}}$$

Perkiraan posisi *K* di dalam larik:

$$I = I_{low} + (I_{upper} - I_{low}) \times \frac{K - K_{low}}{K_{upper} - K_{low}}$$

I_{low} adalah indeks ujung kiri larik
I_{upper} adalah indeks ujung kanan larik

K_{low} adalah elemen minimum di dalam larik (pada indeks I_{low})
K_{upper} adalah elemen maksimum di dalam larik (pada indeks I_{upper})

- Algoritma *interpolation search* sama dengan algoritma *binary search*, hanya mengganti

$$mid \leftarrow (i + j) \text{ div } 2$$

dengan

$$mid \leftarrow i + (j - i) * (K - A(i)) / (A(j) - A(i))$$

sesuai dengan rumus perkiraan posisi K di dalam larik:

$$I = I_{low} + (I_{upper} - I_{low}) \times \frac{K - K_{low}}{K_{upper} - K_{low}}$$

```
procedure Interpolationsearch(input A : LarikInteger,  i, j : integer; K : integer; output idx : integer)
{ Mencari elemen bernilai K di dalam larik A[i..j] dengan interpolation search.
  Masukan: larik A sudah terurut menaik, K sudah terdefinisi nilainya
  Luaran: indek lariks sedemikian sehingga A[idx] = K
}
Deklarasi
    mid : integer

Algoritma:
    if i > j then       { ukuran larik sudah 0}
        idx ← –1        { K tidak ditemukan }
    else
        mid ← i + (j – i) *(K – A(i))/ (A(j) – A (i))
        if A(mid) = K then      { K ditemukan }
            idx ← mid           { indeks elemen larik yang bernilai = K }
        else
            if  A(mid) > K  then
                Interpolationesarch(A, i, mid – 1, K, idx)      { cari di upalarik kiri, di dalam larik A[i..mid]}
            else
                Interpolationsearch(A, mid + 1, j, K, idx)      { cari di upalarik kanan, di dalam larik A[mid+1..j}
            endif
        endif
    endif
```

- Kompleksitas algoritma *interpolation search*:

  - Kasus terburuk: $O(n)$, untuk sembarang distribusi data

  - Kasus terbaik: $O(\log \log n)$, jika data di dalam larik terdistribusi *uniform*

8. Mencari median dan *selection problem*.

- *Selection problem*: mencari elemen terkecil ke-*k* di dalam sebuah senarai beranggotan *n* elemen.
- Jika k = 1 → elemen paling kecil (minimum)
- Jika k = n → elemen paling besar (maksimum)
- Jika k = $\lceil n/2 \rceil$ → elemen median

Bagaimana mencari median dari senarai yang tidak terurut namun tidak perlu mengurutkan senarai terlebih dahulu?

Algoritmanya:

1. Lakukan partisi pada senarai seperti proses partisi pada algoritma *Quick Sort* (varian 2). Partisi menghasilkan setengah elemen senarai lebih kecil atau sama dengan *pivot p* dan setengah bagian lagi lebih besar dari *pivot p*.

$$\underbrace{a_{i_1} \cdots a_{i_{s-1}}}_{\leq p} \; p \; \underbrace{a_{i_{s+1}} \cdots a_{i_n}}_{\geq p}$$

2. Misalkan *s* adalah posisi pem-partisian.

   Jika $s = \lceil n/2 \rceil$, maka pivot *p* adalah nilai median yang dicari

   Jika $s > \lceil n/2 \rceil$, maka median terdapat pada setengah bagian kiri

   Jika $s < \lceil n/2 \rceil$, maka median terdapat pada setengah bagian kanan

Contoh 4: Temukan median dari 4, 1, 10, 9, 7, 12, 8, 2, 15.

Pada contoh ini, $k = \lceil 9/2 \rceil = 5$, sehingga persoalannya adalah mencari elemen terkecil ke-5 di dalam senarai.

Partisi senarai dengan memilih elemen pertama sebagai *pivot*:

4 1 10 9 7 12 8 2 15 (indeks larik dari 1 sampai 9)

Hasil partisi:

2 1 **4** 9 7 12 8 10 15

Karena $s = 3 < 5$, kita memproses setengah bagian kanan:

9 7 12 8 10 15

8 7 **9** 12 10 15

Karena $s = 6 > 5$, kita memproses setengah bagian kiri:

8 7

7 **8**

Sekarang $s = k = 5$ → stop. Jadi median = 8

- Kompleksitas algoritma:

$$T(n) = \begin{cases} a & ,n = 1 \\ T(n/2) + cn & ,n > 1 \end{cases}$$

- Solusi dari relasi rekurens tersebut adalah (dengan menggunakan Teorema Master):

T(n) = T(n/2) + cn = … = O(n)

Latihan Soal
Decrease and Conquer

IF2211 Strategi Algoritma

Soal UTS 2019

Terdapat sebuah matriks A berukuran $n \; x \; n$, yang sudah terurut menaik elemen-elemennya, sedemikian sehingga $A[i][j] < A[i][j']$ untuk $j < j'$; dan $A[i][j] < A[i'][j]$ untuk $i < i'$. Persoalan yang akan diselesaikan adalah menentukan apakah sebuah elemen x ada pada matriks tersebut. Gunakan pendekatan Decrease and Conquer untuk menyelesaikan persoalan tersebut.

(a) Tuliskan langkah-langkah pendekatan yang anda usulkan, dan tuliskan apakah pendekatan tersebut termasuk *decrease by a contant*, *decrease by a constant factor*, atau *decrease by variable size*. Tentukan juga kompleksitas pendekatan usulan anda dalam notasi Big O. **(Nilai 10)**

(b) Terapkan pendekatan usulan anda (langkah per langkah) untuk mencari apakah $x = 29$ terdapat pada matriks berikut ini, dan hasilkan posisi ditemukannya elemen tersebut. **(Nilai 6)**

$$\begin{bmatrix} 10 & 20 & 30 & 40 \\ 15 & 25 & 35 & 45 \\ 27 & 29 & 37 & 48 \\ 32 & 33 & 39 & 50 \end{bmatrix}$$

Jawaban:

(a)

Alternatif I:

Penerapan binary search di tiap baris pada matriks. Setiap baris, periksa elemen tengah (e):

(i) jika e<x maka periksa elemen sebelah kanan dari e, dan abaikan elemen kiri dari e;

(ii) Jika e>x maka periksa elemen sebelah kiri dari e, dan abaikan elemen kanan dari e;

(iii) Jika e=x maka elemen yang dicari ditemukan.

Ini termasuk *decrease by variable*. Binary search pada tiap baris memerlukan waktu O(log n), dan jika diterapkan pada n baris maka kompleksitas waktunya adalah :

O (n log n)

Alternatif II:
Pemeriksaan dimulai dari posisi kanan atas matriks, misal nilai elemennya adalah e
(i) Jika e < x, maka seluruh baris pasti lebih kecil dari x, oleh karena itu abaikan seluruh baris, dan periksa baris berikutnya pada kolom yang sama.

(ii) Jika e > x, maka abaikan seluruh kolom tersebut, karena nilai di kolom tersebut pasti lebih besar dari x, periksa kolom berikutnya (1 kolom sebelah kiri) pada baris yang sama.

(iii) Jika e = x maka x ditemukan pada baris dan kolom tersebut.

Untuk setiap langkah, maka seluruh baris atau kolom diabaikan, jadi pendekatannya adalah decrease by a constant (n), dan kompleksitas waktunya adalah O(n).

(b) Dengan pendekatan Alternatif II:

(i). Periksa elemen [1][4], nilainya adalah 40, dan 40 > 29, maka abaikan seluruh kolom 4, dan bergerak ke 1 kolom di sebelah kiri, yaitu posisi [1][3].

(ii). Nilai pada [1][3] adalah 30, dan 30>29, maka abaikan seluruh kolom 3, dan bergerak 1 kolom ke sebelah kiri, yaitu posisi [1][2].

(iii). Nilai pada posisi [1][2] adalah 20, dan 20 < 29, maka abaikan seluruh baris 1, dan bergerak 1 baris setelahnya, sehingga berada pada posisi [2][2].

(iv). Nilai pada posisi [2][2] adalah 25, dan 25 < 29, maka abaikan seluruh baris 2, dan bergerak ke baris 3 sehingga berada pada posisi [3][2].

(v). Nilai pada posisi [3][2] adalah 29, dan nilai x yang dicari adalah 29. Solusi ditemukan pada posisi [3][2].

Soal UTS 2021

- Terdapat beberapa algoritma untuk mencari pembagi bilangan terbesar (*Greatest Common Divisor*/ GCD) yang sudah anda pelajari. Jelaskan salah satu algoritma untuk mencari GCD dari dua buah bilangan bulat positif, yang menggunakan pendekatan *Decrease and Conquer*. Jelaskan langkah-langkah nya (bukan *pseudo code*), dan terapkan pada contoh mencari GCD(20,12) dan GCD(10,15).

Jawaban:

Dengan algoritma Euclidean yang sudah dipelajari di kuliah Matdis. Misal parameter pertama *m*, parameter kedua n ($m \geq n$).

GCD(*m*, *n*):

if $n = 0$,

retun *m* // GCD(m, 0) = m (basis)

else

$r \leftarrow m$ **mod** *n*

return GCD(*n*, *r*) (rekurens)

endif

- Bilangan *m* pada parameter di setiap langkah tersebut akan terus berkurang (*decrease*), dan ini bagian yang di 'conquer'.
- GCD(20,12) = GCD(12,8) = GCD(8,4) = GCD(4,0) = 4
- GCD(10,15) = GCD(15,10) = GCD(10,5) = GCD (5,0) = 5

UTS 2021

- Diberikan sebuah larik A berukuran n = 9 elemen. Elemen-elemen larik A adalah sebagai berikut: [4, 1, 10, 9, 7, 12, 8, 2, 15]. Kita akan mencari elemen terbesar di dalam larik tersebut dengan metode *decrease and conquer* (memanfaatkan algoritma partisi di dalam Quicksort versi 2). Tuliskan susunan elemen-elemen larik kondisi terakhir setelah ditemukan elemen terbesar .,

UTS 2018

Diberikan larik (*array*) sebagai berikut:

13, 9, 18, 6, 8, 11, 15, 7, 12

Perlihatkan proses mencari elemen terbesar ke-5 dengan algoritma *decrease and conquer* dan memanfaatkan algoritma partisi dari algoritma *Quicksort* varian kedua. *Pivot* yang diambil selalu elemen pertama larik.

UTS 2016

Misalkan anda diberikan sebuah larik bilangan bulat yang terurut. Setiap nilai muncul dua kali, kecuali sebuah nilai tertentu yang hanya muncul sekali. Tugas anda adalah mencari nilai integer yang muncul hanya sekali.

Contoh larik:

(i) 1, 1, 2, 2, **3,** 4, 4, 5, 5, 6, 6, 7, 7, 8, 8

(ii) 10, 10, 17, 17, 18, 18, 19, 19, 21, 21, **23**

(iii) **1,** 3, 3, 5, 5, 7, 7, 8, 8, 9, 9, 10, 10

(angka yang dicetak tebal adalah nilai yang hanya mucul sekali)

(a) Jika diselesaikan dengan algoritma *Brute Force*, bagaimana caranya? (jawaban bukan dalam *pseudo-code*). Berapa kompleksitasnya dalam notasi O-besar?

(b) Jika diselesaikan dengan algoritma *decrease and conquer*, bagaimana langkah-langkahnya? (jawaban bukan dalam pseudo-code). Ilustrasikan langkah-langkh anda dengan contoh larik pertama. Berapa kompleksitas algoritmanya dalam notasi O-besar?

UTS 2014

(a) Tinjau algoritma *interpolation search* untuk mencari elemen bernilai *x* di dalam sebuah larik yang sudah terurut menaik. Tuliskan penurunan rumus estimasi posisi nilai *x* di dalam larik tersebut.

(b) Diberikan sebuah larik berisi elemen-elemen kunci (tidak ada yang sama) yang telah terurut menaik sebagai berikut: [4, 12, 14, 18, 25, 38, 41, 50]. Perlihatkan tahap-tahap pencarian elemen *x* = 41 di dalam larik tersebut dengan menggunakan algoritma *interpolation search*!

(UTS 2020)

Lengkapi tabel berikut ini sesuai petunjuk di tiap soal.

(a) Isikan perbandingan antara ketiga teknik dalam tabel berikut ini. (9)

Aspek	Binary Search	Interpolation Search	Pencarian Median (Selection Problem dengan k = $\lceil n/2 \rceil$)
Decrease by :			
Larik harus terurut (Ya/ Tidak)			
Kompleksitas Algoritma (Big O):			

(b) Terdapat sebuah larik unik A sebagai berikut: A = [3,14, 27, 31, 39, 42, 55, 70, 74, 81, 85, 93, 98]. Carilah indeks di mana nilai K = 85 berada (indeks larik dimulai dari indeks 1), dengan pendekatan Binary Search dan Interpolation Search. Jika tidak ditemukan bilangan tersebut pada larik, pencarian menghasilkan -1. Tuliskan proses pencarian dengan melengkapi tabel berikut ini (penentuan nilai mid dan iterasi untuk tiap jenis pencarian). (11)

	Binary Search			Interpolation Search		
Formula pencarian indeks mid:	mid = …			mid = …		
Iterasi	Indeks awal	Indeks akhir	Indeks mid	Indeks awal	Indeks akhir	Indeks mid
1	1	13	…	1	13	…
2	…	…	…	…	…	…
Dst…	…	…	…	…	…	…
Indeks Akhir hasil pencarian:	…			…		
Jumlah Iterasi	…			…		

(UTS 2019)

Terdapat sebuah matriks A berukuran $n \; x \; n$, yang sudah terurut menaik elemen-elemennya, sedemikian sehingga $A[i][j] < A[i][j']$ untuk $j < j'$; dan $A[i][j] < A[i'][j]$ untuk $i < i'$. Persoalan yang akan diselesaikan adalah menentukan apakah sebuah elemen x ada pada matriks tersebut. Gunakan pendekatan Decrease and Conquer untuk menyelesaikan persoalan tersebut.

(a) Tuliskan langkah-langkah pendekatan yang anda usulkan, dan tuliskan apakah pendekatan tersebut termasuk *decrease by a contant*, *decrease by a constant factor*, atau *decrease by variable size*. Tentukan juga kompleksitas pendekatan usulan anda dalam notasi Big O. **(Nilai 10)**

(b) Terapkan pendekatan usulan anda (langkah per langkah) untuk mencari apakah $x = 29$ terdapat pada matriks berikut ini, dan hasilkan posisi ditemukannya elemen tersebut. **(Nilai 6)**

$$\begin{bmatrix} 10 & 20 & 30 & 40 \\ 15 & 25 & 35 & 45 \\ 27 & 29 & 37 & 48 \\ 32 & 33 & 39 & 50 \end{bmatrix}$$

TAMAT