

ANALISIS ALGORITMA

JIMMY TIRTAWANGSA

<mailto://jimmytirtawangsa@telkomuniversity.ac.id>

<https://lms.telkomuniversity.ac.id>

<https://meet.google.com/gft-oqmz-ghi>

PROGRAM PASCA SARJANA INFORMATIKA
FAKULTAS TEKNIK INFORMATIKA
UNIVERSITAS TELKOM

Semester Genap 2021-2022

Strategi Algoritma

bagian 3 dari 3

- Sorting Lainnya

Metoda Pengurutan Cepat Lainnya

https://en.wikipedia.org/wiki/Sorting_algorithm

Garis Besar Algoritma QuickSort

```
proc QuickSort( A[left..right] )  
  if left < right then  
    p := Pivot(A[left..right])  
    pv := A[p]  
    A[left] ⇔ A[p]  
    k := QuickSplit(A[left+1..right], A[left])  
    A[left] ⇔ A[k]  
    Quicksort(A[left..k-1])  
    Quicksort(A[k+1..right])  
  endif
```

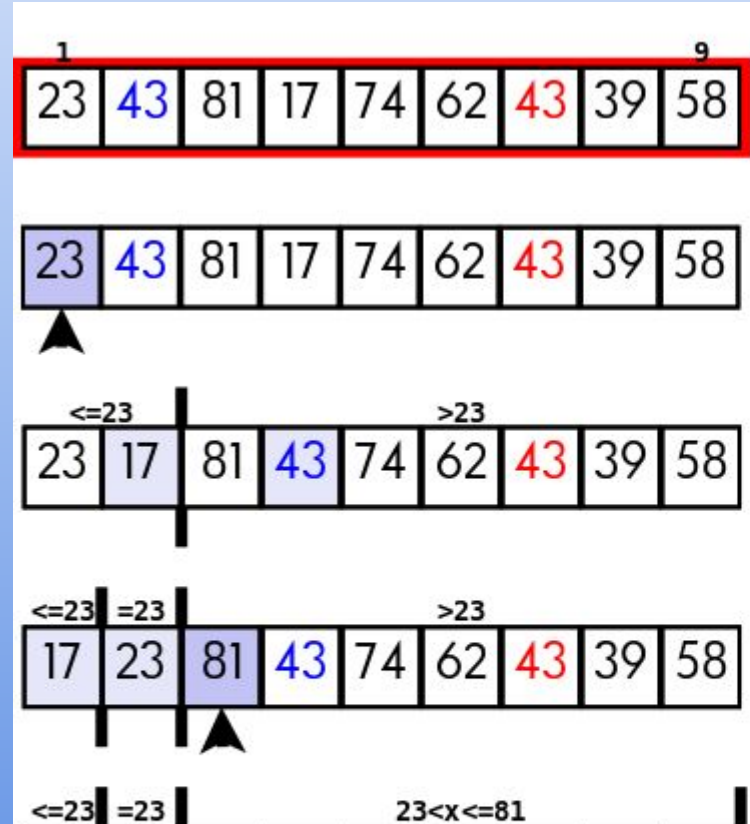
- Pilih suatu data acuan (pivot)
- Bagi data dalam 2 kelompok
 - Data kelompok kiri tidak lebih besar dari p
 - Kelompok kanan lebih besar dari p
 - Data di kelompok kiri bernilai lebih kecil dari data di kelompok kanan
- Ulangi proses yang sama terhadap masing² kelompok
- Gabungan kedua kelompok membentuk data terurut

Pemilihan Pivot QuickSort

- Idealnya adalah nilai median, tetapi median mudah dicari jika sudah terurut
 - Mungkinkah mencari median tanpa mengurutkan data lebih dulu?
- Jika diketahui data terdistribusi uniform, semua data mempunyai kesempatan yang sama sebagai median
 - Ambil saja nilai tengah dari kumpulan data tersebut
 - Ambil saja secara acak salah satu nilai dari kumpulan data tersebut?
- Kalau begitu, bisa saja ambil data yang pertama (atau terakhir) dari list
 - Sederhana kan?
 - Adakah kekurangan/kelemahan pilihan ini yang perlu diperhatikan?

Membagi dua data dalam QuickSort

```
func QuickSplit( A[left..right], pv )
  i := left
  k := right
  while i < k do
    while i ≤ k and A[i] ≤ pv do
      i := i + 1
    endwhile
    while i ≤ k and A[k] > pv do
      k := k - 1
    endwhile
    if i < k then
      A[i] ⇔ A[k]
    endif
  endwhile
  return k
```



QuickSort

```
proc QuickSort( A[left..right] )
  if left < right then
    p := Pivot(A[left..right])
    pv := A[p]
    A[left]  $\Leftrightarrow$  A[p]
    i := left + 1
    k := right
    while i < k do
      while i  $\leq$  k and A[i]  $\leq$  pv do
        i := i + 1
      while i  $\leq$  k and A[k] > pv do
        k := k - 1
      if i < k then
        A[i]  $\Leftrightarrow$  A[k]
      endwhile
    endwhile
    A[left]  $\Leftrightarrow$  A[k]
    Quicksort(A[left..k-1])
    Quicksort(A[k+1..right])
  endif
```

- Pembuktian dengan induksi
- Loop invariant
 - Rekursif : ...
 - Loop luar: ...
 - Loop dalam: ...

QuickSort, LI

- Rekursif (**if** left < right **then** ...)
 - L.I.: Data terbagi dalam 3 group dengan relasi: $A[..left-1] \leq A[left..right] < A[right+1..]$
 - Basis, hipotesis, induksi: ...
- Loop luar (**while** i < k **do** ...)
 - L.I.: $A[left..i-1] < A[k+1..right]$
- Loop dalam (**while** i \leq k **and** $A[i] \leq$ pv **do** ...)
 - L.I.: $A[left..i-1] \leq$ pv
- Loop dalam (**while** i \leq k **and** $A[k] >$ pv **do** ...)
 - L.I.: $A[k+1..right] >$ pv

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 23 | 43 | 81 | 17 | 74 | 62 | 43 | 39 | 58 |
|----|----|----|----|----|----|----|----|----|

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 23 | 43 | 81 | 17 | 74 | 62 | 43 | 39 | 58 |
|----|----|----|----|----|----|----|----|----|

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 23 | 17 | 81 | 43 | 74 | 62 | 43 | 39 | 58 |
|----|----|----|----|----|----|----|----|----|

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 17 | 23 | 81 | 43 | 74 | 62 | 43 | 39 | 58 |
|----|----|----|----|----|----|----|----|----|

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 17 | 23 | 81 | 43 | 74 | 62 | 43 | 39 | 58 |
|----|----|----|----|----|----|----|----|----|

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 17 | 23 | 58 | 43 | 74 | 62 | 43 | 39 | 81 |
|----|----|----|----|----|----|----|----|----|

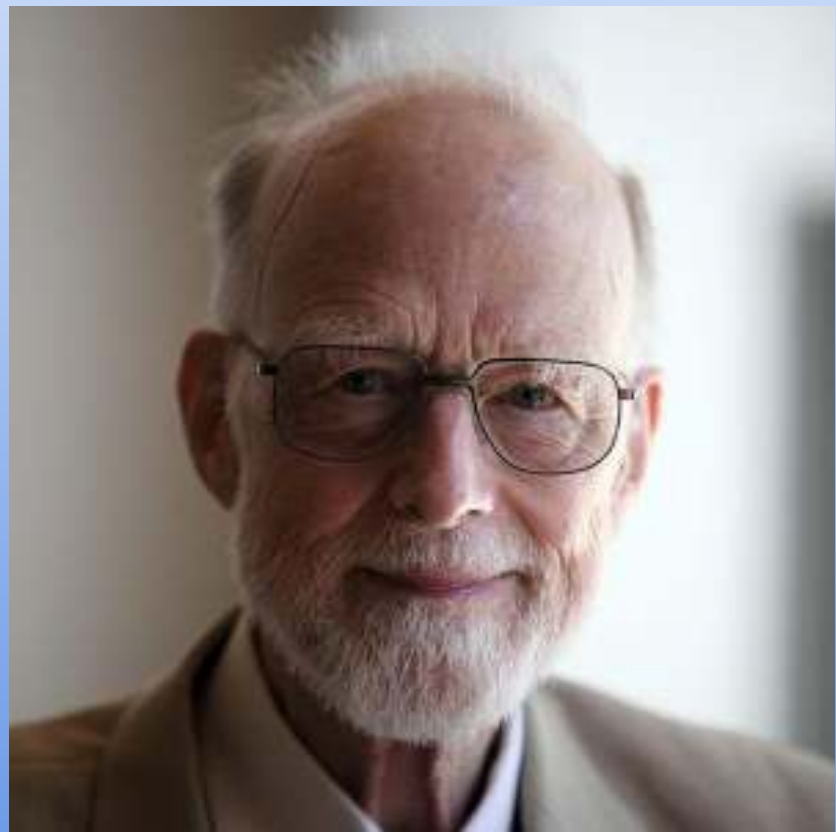
| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 17 | 23 | 58 | 43 | 39 | 43 | 62 | 74 | 81 |
|----|----|----|----|----|----|----|----|----|

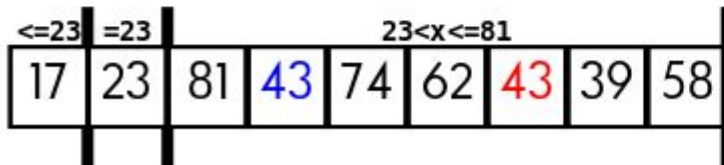
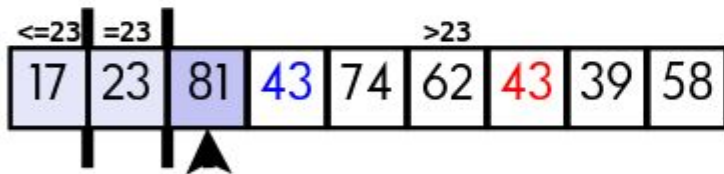
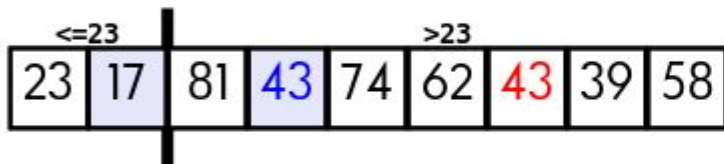
| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 17 | 23 | 43 | 43 | 39 | 58 | 62 | 74 | 81 |
|----|----|----|----|----|----|----|----|----|

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 17 | 23 | 39 | 43 | 43 | 58 | 62 | 74 | 81 |
|----|----|----|----|----|----|----|----|----|

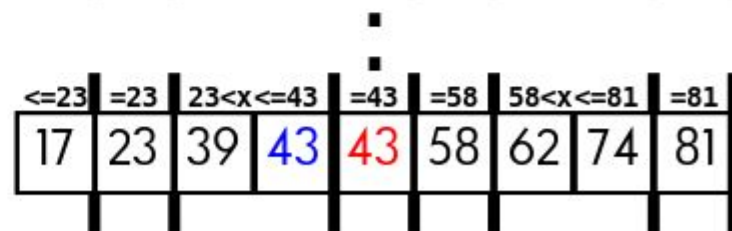
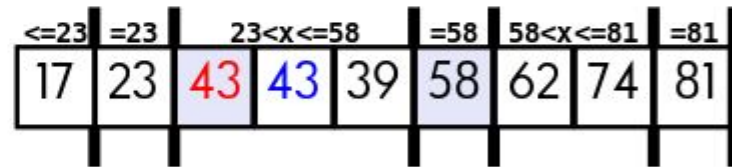
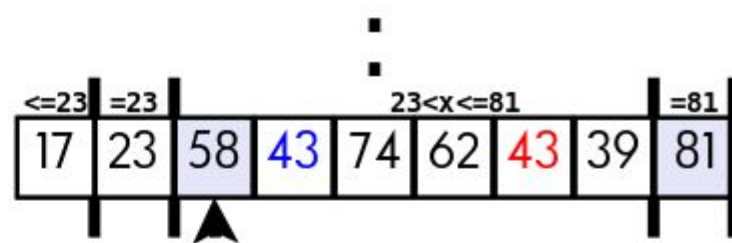
| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 17 | 23 | 39 | 43 | 43 | 58 | 62 | 74 | 81 |
|----|----|----|----|----|----|----|----|----|

C.A.R Hoare menciptakan Quicksort (1959)

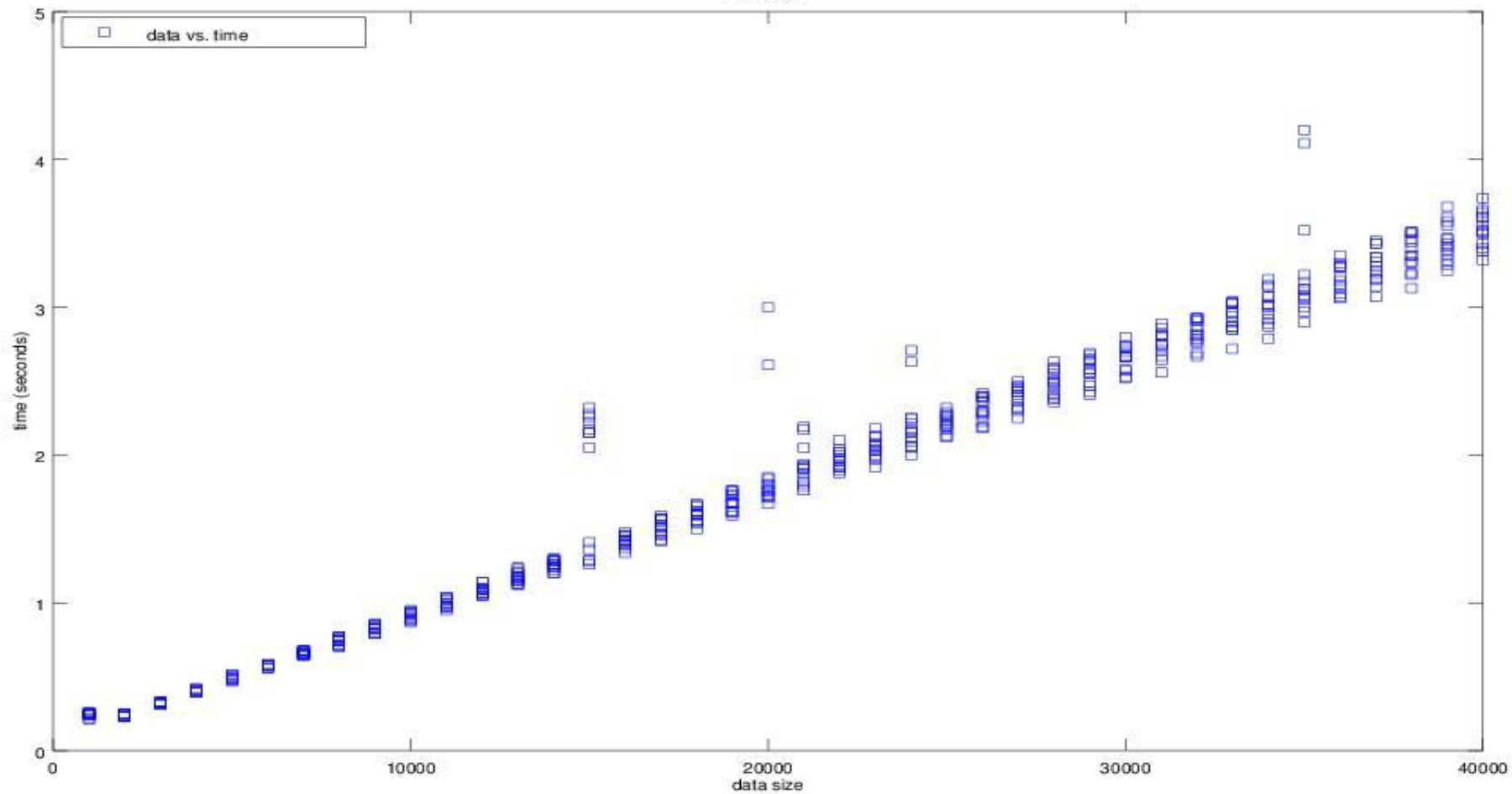




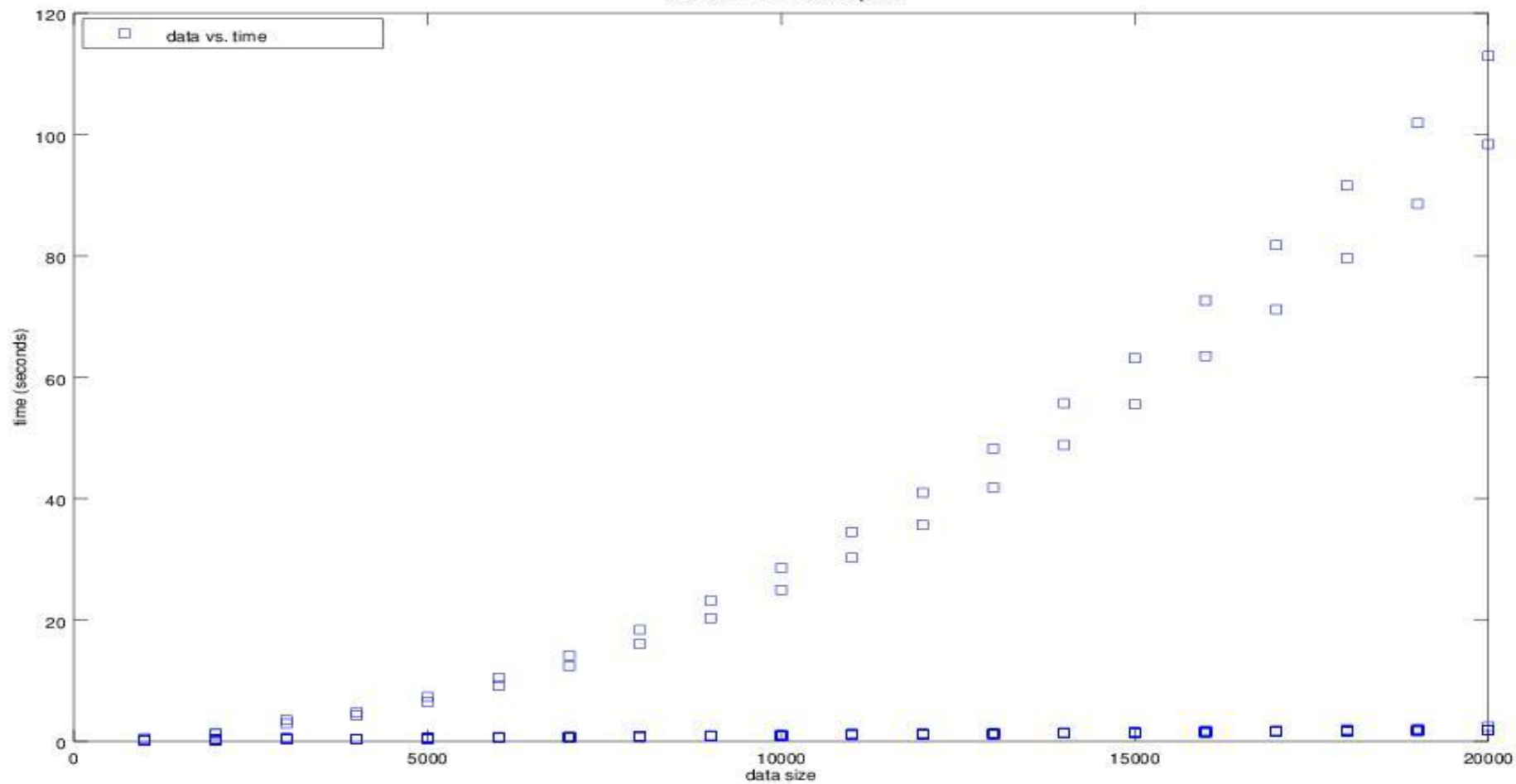
⋮



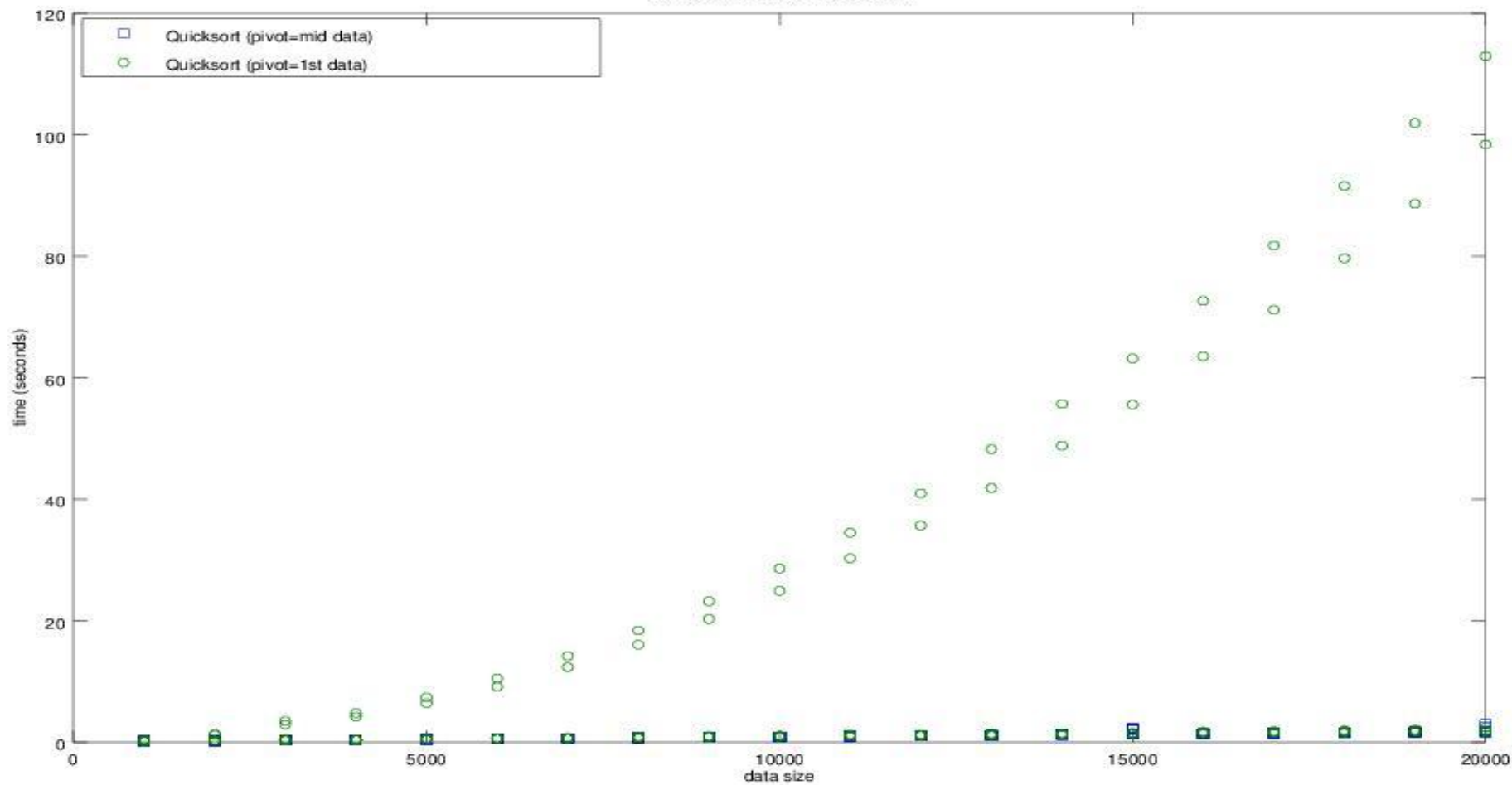
Quicksort



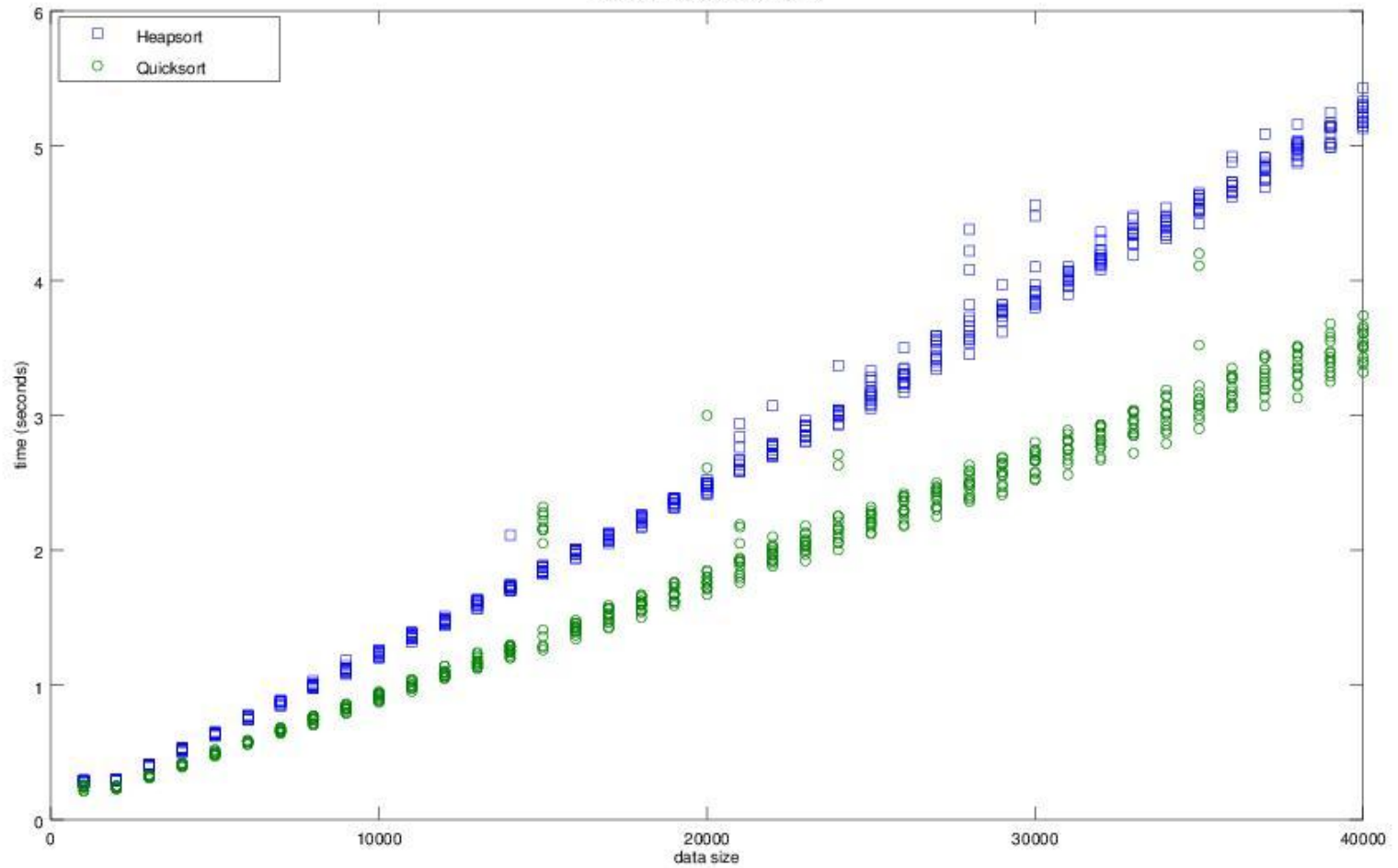
Quicksort w/ 1st data as pivot



Behaviour due to Pivot Selection



Characteristic Fast Algorithms



Analisis Waktu QuickSort

- Misalkan $A[1..n]$

- $$T_{\text{QSORT}}(n) = \begin{cases} c_1 & ; n == 1 \\ T_{\text{QSORT}}(k-\text{left}) + T_{\text{QSORT}}(\text{right}-k) + c_2 n & ; n > 1 \end{cases}$$

- Faktor k mempengaruhi $T_{\text{QSORT}}(n)$

```
proc QuickSort( A[left..right] )  
  if left < right then  
    p := Pivot(A[left..right])  
    pv := A[p]  
    A[left]  $\Leftrightarrow$  A[p]  
    k := QuickSplit(A[left+1..right], pv)  
    A[left]  $\Leftrightarrow$  A[k]  
    Quicksort(A[left..k-1])  
    Quicksort(A[k+1..right])  
  endif
```

Analisis Waktu QuickSort

- Jika k-left atau right-k \approx konstan c_3 , maka

$$T_{\text{Qsort}}(n) = T(c_3) + T(n-c_3) + c_2 n \approx (c_2 n) \cdot (n/c_3) = (c_2 n)^2 / c_3$$

Ada n/c_3 iterasi

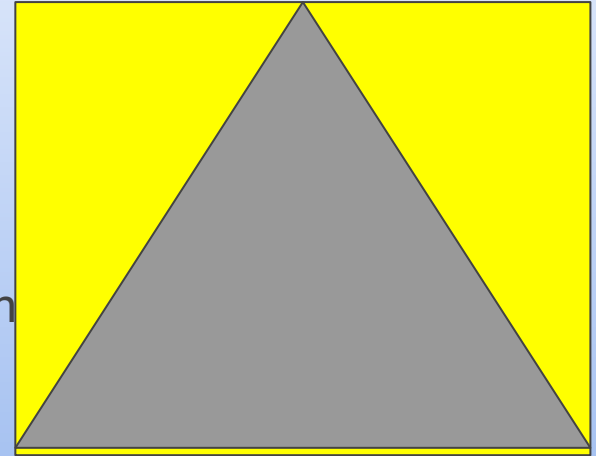
- Jika k-left dan right-k $\approx 1/2 n$, maka

$$T_{\text{Qsort}}(n) = 2T(1/2 n) + c_2 n \approx (c_2 n) \cdot \lg n = O(n \lg n)$$

Tinggi pohon ideal dengan n leaf

- Jika k-left dan right-k $\approx c_4 n$, $0 < c_4 < 1$ maka

$$T_{\text{Qsort}}(n) = T(c_4 n) + T((1-c_4)n) + c_2 n \approx (c_2 n) \cdot (\lg n / \lg c_4) \approx c_5 n \lg n = O(n \lg n)$$



Menuju Waktu Rerata: Randomization QuickSort

```
func Pivot( A[left..right] )  
    return := (left + right) / 2
```

```
func Pivot( A[left..right] )  
    return := left
```

```
func Pivot( A[left..right] )  
    return := Random(left, right)
```

Insertion sort: n^2 n^2 n
Quicksort : n^2 $n \cdot \lg(n)$ $n \cdot \lg(n)$

Algoritma Median

- Algoritma cepat pencarian median untuk pivot, $O(n)$
- Modifikasi algoritma quicksort untuk mencari median atau data posisi ke- i

Akhir dari Topik Minggu 03