Fakultas Informatika
School of Computing
Telkom University
Think Big! Go Forward!

Telkom University

ANALISIS ALGORITMA

Week 04: Batas Bawah Kompleksitas Problem Komputasi

PROGRAM PASCA SARJANA INFORMATIKA
FAKULTAS TEKNIK INFORMATIKA
UNIVERSITAS TELKOM

2022/2023

Batas Bawah Kompleksitas

bagian 1 dari 2

- Batas bawah problem komputasi
- Solusi kasus khusus

Analisis Batas Bawah Problem Komputasi

Analisis Problem Komputasi

Untuk besar data sebanyak n, berapa komputasi esensial harus dilakukan?

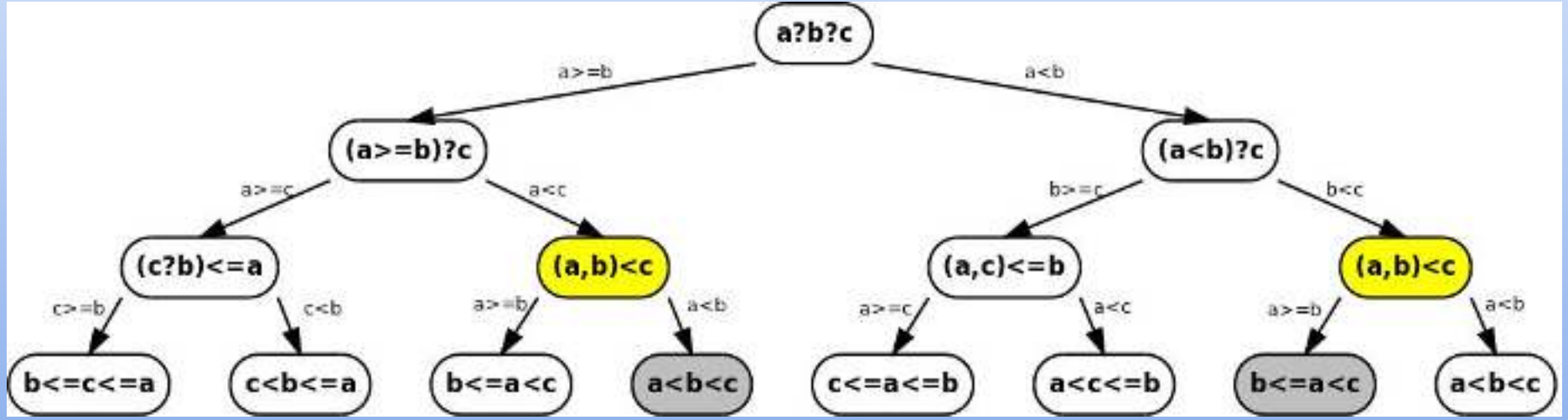
Contoh:

- Selection sort jumlah perbandingan ada sebanyak $O(n^2)$
- Beberapa **mungkin redundant**, tetapi harus dilakukan pada selection sort
- Pada heapsort jumlah perbandingan berkurang menjadi sebanyak $O(n \lg n)$
- Tetapi beberapa perbandingan **yang sama diyakini pasti masih terjadi**
- Pada problem sorting, berapa paling sedikit perbandingan harus dilakukan?
- → Kurang dari itu akan **ada suatu dataset yang tidak mungkin** diurutkan

Manfaat Pohon Keputusan Problem Sorting

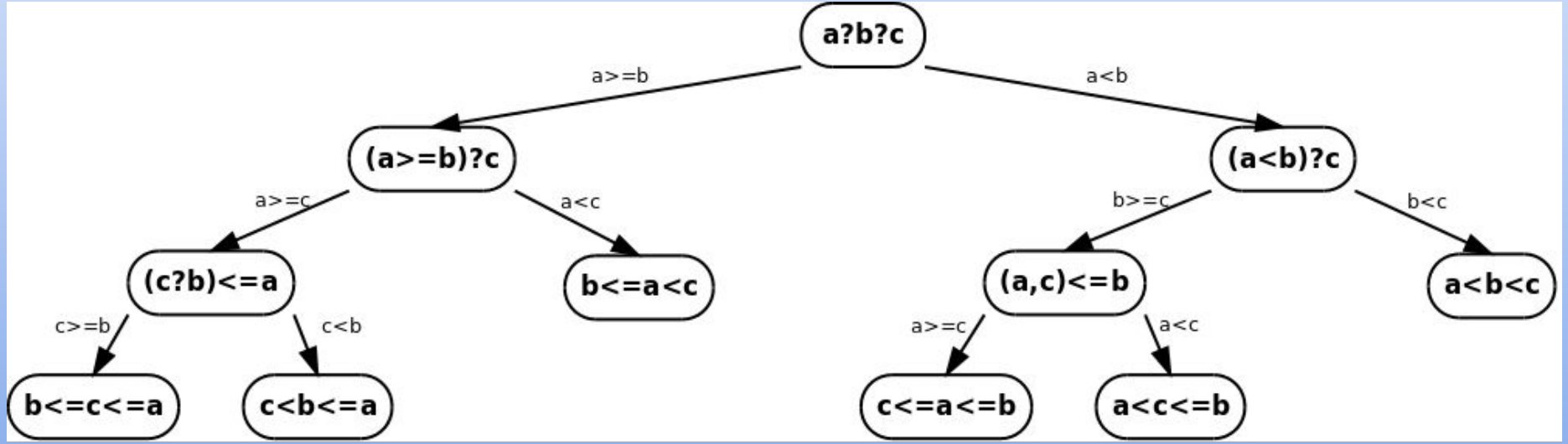
- Visualisasi alur proses pembandingan antara dua data
- Membentuk pohon biner
- Percabangan pada node merupakan hasil apakah $x<y$, atau $x \geq y$
- Leaf merupakan **semua** permutasi dari data semula
  - **jika pohon keputusan yang dibangun tidak menghasilkan semua permutasi, maka . . .**
- Jalur dari root ke leaf merupakan rangkaian operasi perbandingan yang digunakan untuk mencapai satu permutasi tersebut.
- Maka **tinggi pohon menjadi representasi kompleksitas** sorting tersebut

Contoh Pohon u/ **Selection Sort 3 Data**



- Ada beberapa operasi yang duplikat?
- Untuk 3 data, diperlukan 3x perbandingan untuk mencapai semua kemungkinan permutasi data
- Untuk 3 data, ada berapa kemungkinan permutasi data? 😕

Pohon Keputusan Optimal



- Tidak semua perbandingan diperlukan
- Untuk menggapai semua kemungkinan permutasi:
  Minimum kebutuhan diukur terhadap jumlah operasi perbandingan terbanyak

Analisis Batas Bawah **Problem Komputasi**

- Berapa banyak kebutuhan minimum operasi komparasi?
- Kedalaman dari pohon keputusan dapat menunjukan jumlah operasi komparasi minimum yang diperlukan suatu proses (sorting)
- Jumlah permutasi untuk n data adalah p(n) = . . .
  - misal 3 data: (a,b,c) (a,c,b) (b,a,c) (b,c,a) (c,a,b) (c,b,a) ⇒ ada sebanyak . . .
- Tinggi pohon biner dengan p(n) leaf adalah h(p(n)) = . . .

Analisis Batas Bawah **Problem Komputasi**

- Berapa banyak kebutuhan minimum operasi komparasi?
- Kedalaman dari pohon keputusan dapat menunjukan jumlah operasi komparasi minimum yang diperlukan suatu proses (sorting)
- Jumlah permutasi untuk n data adalah $p(n) = n!$
  - misal 3 data: (a,b,c) (a,c,b) (b,a,c) (b,c,a) (c,a,b) (c,b,a) ⇒ ada sebanyak 3!=3.2=6
- Tinggi pohon biner dengan p(n) leaf adalah $h(p(n)) = \ell g(n!) =$ . . . ?
  - Ingat $n! < n^n = n.n. . . n$
  - Sehingga $\ell g(n!) < \ell g(n^n) = n\ell g(n)$
  - Tapi ini berarti kebutuhan maksikum adalah $T_{SORT}(n) = h(p(n)) = O(n\ell g(n))$
  - Sedangkan yang dicari adalah kebutuhan minimum komputasi: $\mathbf{T_{SORT}(n) = \Omega(....)}$

Batas Bawah Kompleksitas Problem Sorting

$T_{SORT} = \Omega(\ell g(n!))$

$\frac{1}{2}n^{n/2} \leq n! \leq n^n$

$T_{SORT} \geq c\ell g(\frac{1}{2}n^{n/2})$; c positif

$= \frac{1}{2}cn.(\ell g(n)-\ell g(2))$

$= \frac{1}{2}cn\ell g(n) + -\frac{1}{2}cn$

$= \frac{1}{4}cn\ell g(n) + (-\frac{1}{2}cn + \frac{1}{4}cn\ell g(n))$

$\geq \frac{1}{4}cn\ell g(n)$; $n \geq 4$

$T_{SORT} = \Omega(n\ell g(n))$; untuk $n \geq 4$

$(-\frac{1}{2}cn + \frac{1}{4}cn\ell g(n)) \geq 0$

$\frac{1}{4}cn\ell g(n) \geq \frac{1}{2}cn$

$n\ell g(n) \geq 2n$

$\ell g(n) \geq 2$

atau **$n \geq 4$**

Konsekuensi

- Dengan komparasi tidak ada metoda sorting lebih baik dari $\Omega(n\lg(n))$
- Suatu struktur data yang terbentuk dengan metoda komparasi
  - tidak dapat memberi efek sorting dalam waktu lebih cepat
    - Ingat preproses untuk membuat struktur heap ↛ data terurut
    - Solusi problem lain memberikan efek keterurutan data, i.e. problem lain dapat dipakai untuk mengurutkan data → Implisit min cost bagi problem lain tersebut.
  - membatasi kompleksitas beberapa operasi dalam struktur
    - Jika operasi keseluruhan menjadikan terurut → ada batas minimum cost per operasi (ttg. BST dan keterurutan data)
- Sorting dapat lebih cepat bila tidak menggunakan operasi komparasi
  - Ada batasan kemampuan, atau hanya untuk kasus data tertentu
  - Perlu tahu karakteristik domain data

Akhir Bagian 1 dari Topik Minggu 04