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

Telkom University

ANALISIS ALGORITMA

Week 03: Strategi Algoritma

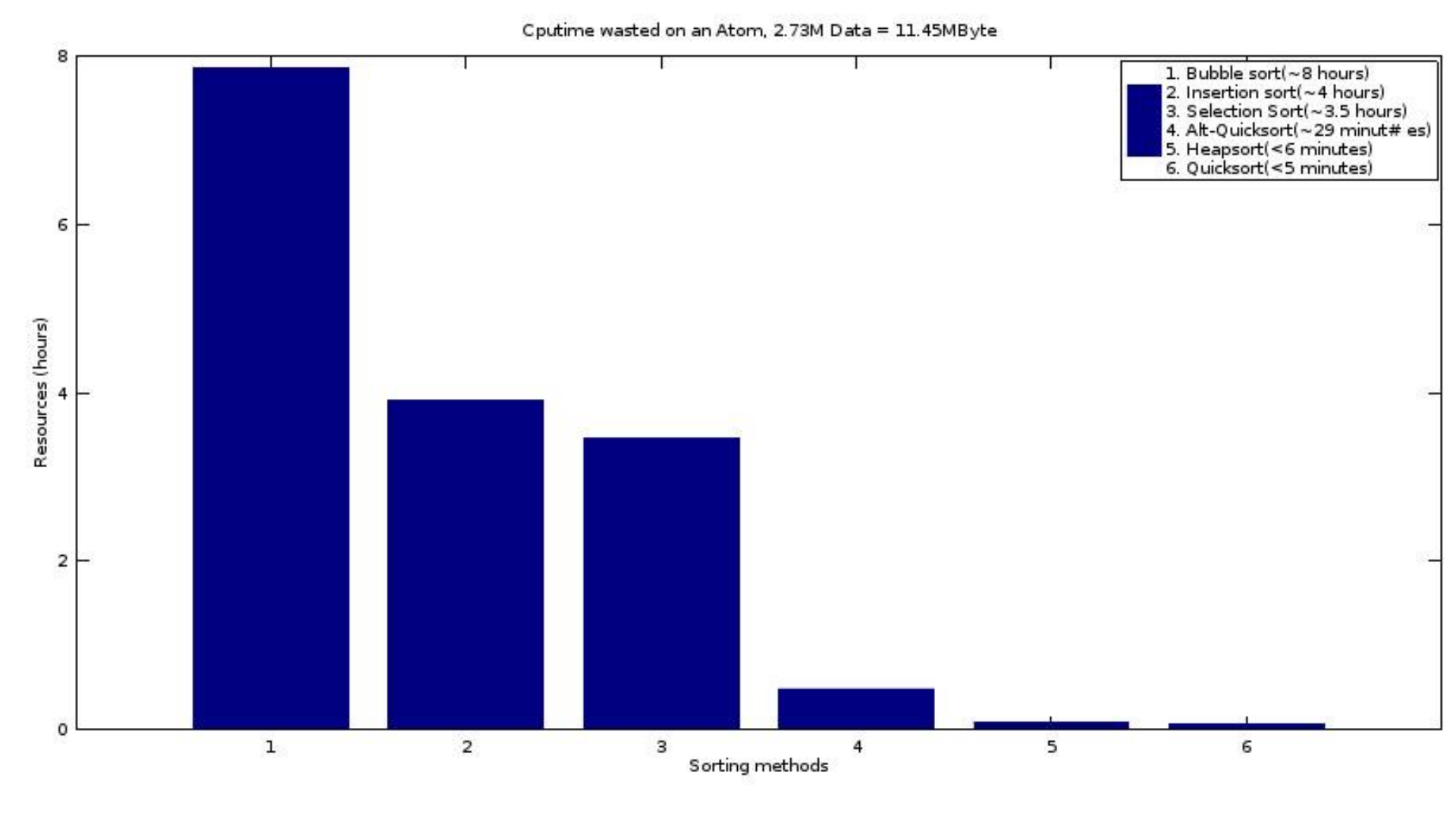
PROGRAM PASCA SARJANA INFORMATIKA
FAKULTAS TEKNIK INFORMATIKA
UNIVERSITAS TELKOM

2022/2023

Strategi Algoritma

bagian 1 dari 3

- Strategi penyempurnaan solusi problem komputasi
- Kasus Sorting


Cputime wasted on an Atom, 2.73M Data = 11.45MByte
1. Bubble sort(~8 hours)
2. Insertion sort(~4 hours)
3. Selection Sort(~3.5 hours)
4. Alt-Quicksort(~29 minut# es)
5. Heapsort(<6 minutes)
6. Quicksort(<5 minutes)
Resources (hours)
0
2
4
6
8
1
2
3
4
5
6
Sorting methods

Strategi Penyempurnaan Solusi Komputasi

Karakteristik lain algoritma yang berguna

Cepat/memori kecil dll. bukan satu[2]nya properti yang diperhitungkan dalam pengembangan algoritma

- Cepat tapi salah, tidak berguna
- Stabilitas data, tidak mengubah data/posisi yang tidak perlu diubah
- Konsistensi waktu, pada situasi realtime
- Untuk jumlah data besar, lokasi dan variasi tipe
- Solusi eksak atau cukup aproksimasi (approx. sorting)
- Kompatibilitas (variasi format, dll.)

Penyempurnaan Efisiensi Solusi Komputasi

- Cari cara untuk menghindari **komputasi yang tidak perlu**
- Hindari **komputasi yang sama dengan parameter yang sama**, karena akan memberikan hasil yang sama
- Metoda algoritma mengarah pada penyempurnaan efisiensi diatas, misal:
  - Divide and conquer, Pruning
  - Dynamic programming
  - Greedy approach
  - Branch and bound
- Penting untuk analisis mendalam atas karakteristik dari problem tersebut dan/atau informasi tambahan tentang problem tersebut

Pangkas komputasi yang tidak perlu

- Proses pencarian (searching) harus satu per satu (brute force) jika tidak ada informasi tambahan
- Proses binary search dapat dilakukan karena diketahui data dalam keadaan terurut
  - Setiap satu pengujian, bukan hanya satu data yang disingkirkan
  - Ada sebagian data yang otomatis diketahui tidak akan mempengaruhi proses pencarian
  - Dan dapat dipangkas/diabaikan pada iterasi berikutnya

Komputasi berulang

- Pada algoritma selection sort, misalkan dengan data contoh
  - 2 3 5 7 11 13
- Pada iterasi 1, komparasi yang dilakukan (mencari Max):
  - 2↔3, 3↔5, 5↔7, 7↔11, 11↔13
- Pada iterasi 2 sebagian besar komparasi yang sama ternyata diulang:
  - 2↔3, 3↔5, 5↔7, 7↔11
- Dst nya sampai iterasi terakhir, pada dasarnya komparasi yang sudah dilakukan sebelumnya akan diulang
- Ide baru perlu dicari agar redundansi ini **dapat dihilangkan/dikurangi**

Divide and Conquer, Pemangkasan

- Divide and conquer, komputasi dikurangi dengan tidak melakukan untuk seluruh data yang ada pada setiap iterasi
  - Quicksort
  - Mergesort
- Pruning, komputasi dikurangi dengan memastikan sebagian data tidak perlu diproses lagi pada iterasi berikutnya
  - Pencarian median dalam waktu linear

Dynamic Programming

- Jika komputasi utama terdiri dari sejumlah komputasi kecil serupa (terdefinisi rekursif)
- dan Komputasi kecil tersebut mungkin sama dan digunakan beberapa kali
- Pengulangan dapat dihindari dengan **menyimpan hasil komputasi kecil** tersebut untuk kemudian dipakai lagi tanpa harus diproses ulang
- **Komputasi utama cukup dengan mencari hasil komputasi kecil** dari tabel
    - Pilih biaya yang lebih murah antara mencari hasil sebelumnya atau menghitung ulang.
    - Contoh kasus menghitung nilai pada deret fibonacci

Pendekatan Greedy

- Dengan memilih alternatif terbaik dari yang terlihat sejauh horison komputasi yang dilakukan
- Untuk banyak problem, semua alternatif solusi terlihat dengan jelas, sehingga pilihan selalu yang terbaik
  - Selection sort : pada setiap iterasi cari nilai terbesar berikutnya?
- Untuk banyak problem lain, tidak semua alternatif solusi terlihat dalam batas horison komputasi, sehingga pilihan terbaik saat itu belum tentu pada akhirnya yang terbaik
  - Problem Travelling Salesman: pada setiap iterasi cari kota berikutnya yang belum dikunjungi dengan jarak terpendek dari kota yang sekarang dikunjungi?

Branch and bound, Brute force

- Brute force, semua alternatif solusi harus diuji untuk menentukan mana yang terbaik.
  - Mencari nilai maksimum dari sejumlah data tidak terurut
- Branch and bound, jika dalam proses diatas, separuh jalan, suatu alternatif sudah diduga tidak mungkin ada solusi, maka alternatif tersebut ditinggal (bound), dan dilanjutkan dengan menguji alternatif lain (branch)
  - Mencari jalur dari a ke b dengan total biaya tidak lebih dari c. Jika suatu jalur dari a separuh jalan saja sudah mencapai target c, tentunya tidak perlu dilanjutkan. Karena untuk mencapai b, total biaya pasti akan lebih dari c.

Akhir Bagian 1 dari Topik Minggu 03