

Berdoalah terlebih dahulu agar Anda berhasil dalam mengerjakan ujian ini!

A. Algoritma Greedy

1. Di sebuah komputer, terdapat banyak prosesor untuk mengerjakan *task*. Setiap *task* memiliki waktu mulai T dan lama eksekusi D . Satu prosesor dapat menjalankan banyak *task* secara sekuensial asalkan waktunya tidak beririsan ($T_i + D_i \leq T_{i+1}$). Jika terdapat 7 buah *task* $(T_i, D_i) = [(2, 3), (0, 4), (8, 4), (5, 6), (9, 3), (8, 1), (12, 2)]$, dengan algoritma *greedy* tentukan dan jelaskan: (7,5 + 7,5 = 15)
- (a) Berapa jumlah minimal prosesor yang dibutuhkan?
- (b) Jika *task* yang memiliki $D > 4$ dapat dipecah menjadi dua *task* sehingga bisa dieksekusi secara paralel (beda prosesor) dengan ketentuan $(T_i, D_i) \rightarrow [(T_i, D_i \text{ div } 2), (T_i, D_i \text{ div } 2)]$, apakah hasilnya akan lebih optimal?

B. Divide and Conquer

2. Diberikan sebuah larik **tidak terurut** yang berisi $n \geq 2$ bilangan bulat, misalnya $[x(1), \dots, x(n)]$, asumsikan n perpangkatan 2, yaitu $n = 2^k$. Kita ingin menemukan jarak (*step*) terbesar d , yang didefinisikan sebagai nilai maksimum $x(j) - x(i)$ untuk **semua** $j > i$. Misalnya, untuk (5 + 10 + 5 = 20)
- $x = [22, 2, 8, 7, 4, 10, -1, 6]$, maka $d = x(6) - x(2) = 10 - 2 = 8$
- (a) Jika diselesaikan dengan algoritma *brute force*, bagaimana langkah-langkahnya, dan berapa kompleksitas waktunya dalam notasi O-besar?
- (b) Jika diselesaikan dengan algoritma *divide and conquer*, tuliskan langkah-langkahnya (boleh dalam notasi *pseudo-code* yang memperlihatkan tahap *divide*, *conquer*, dan *combine*), tentukan $T(n)$ dalam relasi rekurens, lalu tentukan notasi O-besarnya
- (c) Perlihatkan proses *divide and conquer* untuk menemukan step terbesar pada contoh larik x di atas
3. Misalkan Anda akan memilih satu di antara 3 algoritma berikut:
- (i) Algoritma A memecahkan persoalan berukuran n dengan membaginya menjadi lima upa-persoalan berukuran $n/2$, menyelesaikan setiap upa-persoalan secara rekursif, lalu menggabungkan solusinya dalam waktu linier.
- (ii) Algoritma B memecahkan persoalan berukuran n dengan membaginya menjadi dua upa-persoalan masing-masing berukuran $n - 1$, menyelesaikan setiap upa-persoalan secara rekursif, lalu menggabungkan solusinya dalam waktu konstan.
- (iii) Algoritma C memecahkan persoalan berukuran n dengan membaginya menjadi sembilan upa-persoalan berukuran $n/3$, memecahkan setiap upa-persoalan secara rekursif, lalu menggabungkan solusinya dalam waktu kuadrat.
- Tentukan kompleksitas waktu, $T(n)$, untuk setiap algoritma, dan kompleksitas waktu asimptotiknya dalam notasi O-besar, lalu tentukan algoritma mana yang akan Anda pilih dan jelaskan alasannya. (4 + 4 + 4 + 3 = 15)

C. Decrease and Conquer

4. Terdapat sebuah timbangan seperti pada gambar di samping. Timbangan digunakan untuk menentukan satu buah bola dari 9 buah bola, yang memiliki **bobot lebih berat** dibandingkan yang lain. Delapan buah bola yang lain memiliki bobot yang sama. Dengan pendekatan *Decrease and Conquer*, (7,5 + 7,5 = 15)
- (a) Tentukan langkah-langkah penimbangan sebanyak **dua kali** untuk mendapatkan satu buah bola yang lebih berat dibandingkan bola yang lain. Tentukan termasuk



pendekatan *Decrease and Conquer* yang mana (*decrease by constant*, *decrease by constant factor*, atau *decrease by variable size*).

- (b) Jika n merepresentasikan banyaknya bola (dalam kelipatan 3), dan $(n - 1)$ adalah banyaknya bola dengan bobot yang sama, tentukan kompleksitas waktu pendekatan *Decrease and Conquer* pada jawaban (a); kemudian tentukan kompleksitasnya dalam notasi Big-O.

D. DFS, BFS, IDS

5. Terdapat sebuah informasi berupa senarai ketetanggaan sebagai berikut, untuk merepresentasikan suatu graf berarah.

{ A : B,D,
 B : C, F,
 C : E, G, H,
 D : F,
 E : B, F,
 F : A
 H : A }

(5 + 5 + 5 + 5 = 20)

- (a) Gambarkan graf berarah sesuai dengan informasi senarai ketetanggaan tersebut.
- (b) Jika dari simpul A ingin menuju simpul F, **tentukan jalur penelusuran/ pencarian dengan BFS**, dan tentukan **jalur hasil untuk menuju F dari simpul A**. Jika terdapat lebih dari satu simpul tetangga, pemilihan simpul sesuai urutan abjad.
- (c) Jika dari simpul A ingin menuju simpul F, **tentukan jalur penelusuran/ pencarian dengan DFS**, dan tentukan **jalur hasil untuk menuju F dari simpul A**. Jika terdapat lebih dari satu simpul tetangga, pemilihan simpul sesuai urutan abjad.
- (d) Jika dari simpul A ingin menuju simpul F, **tentukan jalur penelusuran/ pencarian dengan DLS** dengan batas kedalaman **3**, dan tentukan **jalur hasil untuk menuju F dari simpul A**. Jika terdapat lebih dari satu simpul tetangga, pemilihan simpul sesuai urutan abjad.
6. Sebuah robot pemadam api memasuki sebuah matriks 4 x 4 (indeks matriks dimulai dari 1). Robot hanya dapat bergerak satu sel ke kiri, ke kanan, ke bawah, atau ke atas. Di dalam matriks terdapat beberapa sel ranjau (robot tidak tahu di dalam sel ada ranjau). Jika robot memasuki sel ranjau, maka robot mati karena ledakan ranjau. Robot mulai berjalan dari sebuah titik masuk (**in**) sampai menemukan sel api tersebut. Urutan gerakan robot dari setiap sel harus konsisten, yaitu kiri, kanan, atas, bawah.
- (a) Gambarkan pohon ruang status lengkap sebelum pencarian
- (b) Gambarkan pohon ruang status pencarian sel yang mengandung api secara DFS. Beri nomor setiap status sesuai urutan pembangkitannya, setiap sisi (cabang) pada pohon ditulis gerakannya (kiri, kanan, atas, bawah). Pada setiap status tulis koordinat elemen matriks yang dimasuki.
- (c) Ulangi soal (b) untuk BFS

(5 + 5 + 5 = 15)

