Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Jawaban Ujian Tengah Semester **IF2211 Strategi Algoritma**
Rabu, 26 Maret 2024
Waktu: 110 menit
Dosen: Rinaldi Munir (K1), Nur Ulfa Maulidevi (K2), Rila Mandala (K3), Monterico Adrian (K3)

Berdoalah terlebih dahulu agar Anda berhasil dalam mengerjakan ujian ini!

Algoritma Greedy

1. Dalam sebuah permainan *adventure*, pemain dapat membawa item *healing* seberat 50 satuan. Terdapat 4 item untuk *healing*: Elixir yang mempunyai berat 28 dan kemampuan *healing* 40, Potion yang mempunyai berat 22 dan kemampuan *healing* 32, Tonic yang mempunyai berat 24 dan kemampuan *healing* 35, dan Ointment yang mempunyai berat 32 dan kemampuan *healing* 45. Jika pemain hanya dapat membawa satu untuk tiap jenis item dan item dapat dibawa secara sebagian, tentukan item mana saja yang dibawa yang akan memberikan keuntungan maksimum dengan algoritma Greedy. **(Nilai: 15)**

Jawaban:
Berdasarkan teorema, algoritma greedy untuk fractional knapsack dengan strategi pemilihan objek berdasarkan pi /wi terbesar menghasilkan solusi yang optimum

Properti objek				
i	w_i	p_i	p_i/w_i	*density*
Elixir	28	40	1,43	1/7
Potion	22	32	1,45	1
Orb	24	35	1,46	1
Ointment	32	45	1,41	0
			Total bobot	50
			Total keuntungan	72,71

Brute Force & Divide and Conquer

2. Diberikan sebuah larik dengan n elemen yang berisi nilai-nilai dalam bilangan riil. Kita akan mencari perbedaan maksimum antara dua elemen di dalam larik. Contoh: A = [4.5, 10, –2, 3.14, –7.25], perbedaan maksimum adalah 17.25. **(Nilai: 15)**
   (a) Jika diselesaikan secara *brute force*, bagaimana algoritmanya dan berapa kompleksitas waktunya?
   (b) Jika diselesaikan secara *divide and conquer*, tuliskan langkah-langkahnya, tuliskan pseudo-code nya, tentukan kompleksitas waktunya dalam relasi rekurens, lalu notasi kompleksitas asimptotiknya.

Jawaban:
Perbedaan maksimum antara dua elemen di dalam larik diperoleh dari selisih nilai maksimum dengan nilai minimum. Cari nilai maks dan min di dalam larik secara brute force atau divide and conquer:

(a) Anggap elemen pertama larik sebagai nilai maks dan nilai min sementara sekaligus, lalu lakukan traversal larik mulai dari elemen kedua sampai elemen ke-n untuk menemukan elemen maks dan min sebenarnya.

```
maks ← A[1]
min ← A[1]
for k ← 2 to n do
    if A[k] > maks then maks ← A[k] endif
    if A[k] < min then min ← A[k] endif
endif
```

selisih_maksimum ← maks – min

Kompleksitas algoritma diukur dari jumlah operasi perbandingan elemen larik adalah T(n) = 2n – 2 = O(n)

(b) Terapkan algoritma MinMaks secara *divide and conquer* yang telah diberikan di dalam kuliah, lalu kurangkan maks dan min yang ditemukan untuk mendapatkan nilai seleisih terbesar.

Prosedur MinMaks2(A, n, min, maks)

Algoritma:

1. Untuk kasus $n = 1$ atau $n = 2$,
   SOLVE: Jika $n = 1$, maka $min = maks = A[n]$
   Jika $n = 2$, maka bandingkan kedua elemen untuk menentukan *min* dan *maks*

2. Untuk kasus $n > 2$,
   (a) DIVIDE: Bagi dua larik *A* menjadi dua bagian yang sama, A1 dan A2, masing2 n/2 elemen

   (b) CONQUER:
   MinMaks2(A1, n/2, min1, maks1)
   MinMaks2(A2, n/2, min2, maks2)

   (c) COMBINE:
   if min1 < min2 then min ← min1 else min ← min2
   if maks1 < maks2 then maks ← maks2 else maks ← maks1

Panggil prosedur MinMaks:

MinMaks(A, n, min, maks)

Selisih_maksimum ← maks – min

Kompleksitas algoritma dihitung dari jumlah perbandingan elemen-elemen larik:

$$T(n) = \begin{cases} 0 & ,n = 1 \\ 1 & ,n = 2 \\ 2T(n/2) + 2 & ,n > 2 \end{cases}$$

Penyelesaian:

Asumsi: $n = 2^k$, dengan k bilangan bulat positif, maka

$$\begin{aligned} T(n) &= 2\mathrm{T}(n/2) + 2 \\ &= 2(2T(n/4) + 2) + 2 = 4T(\mathrm{n}/4) + 4 + 2 \\ &= 4T(2\mathrm{T}(n/8) + 2) + 4 + 2 = 8T(n/8) + 8 + 4 + 2 \\ &= \ldots \\ &= 2^{k-1}\, T(2) + \sum_{i=1}^{k-1} 2^i \\ &= 2^{k-1} \cdot 1 + 2^k - 2 \\ &= n/2 + n - 2 \\ &= 3n/2 \ - 2 \\ &= O(n) \end{aligned}$$

Teorema Master

3. Dengan menggunakan Teorema Master, tentukan notasi notasi Big-Oh untuk kompleksitas waktu dalam bentuk relasi rekurens berikut. Jika teorema master tidak bisa diterapkan jelaskan alasannya: **(Nilai: 10)**

a. $T(n) = 5T(n/4) + \sqrt{n}$
b. $T(n) = T(\sqrt{n}) + 5$ (Petunjuk: misalkan $n = 2^m$)
c. $T(n) = 2T(\sqrt{n}) + 1$ (Petunjuk: misalkan $n = 2^m$)

Jawaban:

(b) $T(n) = 5T(n/4) + n^{1/2}$, Berdasarkan teorema master, a = 5, b = 4 dan d = ½, maka $5 > 4^{1/2}$ (case 3),
T(n) adalah $O(n^{4^{\log 5}})$

(c) $T(n) = T(\sqrt{n}) + 5$, misalkan $n = 2^m$, maka $T(2^m) = T(2^{m/2}) + 5$. Misalkan $S(m) = T(2^m)$,
maka $S(m) = S(m/2) + 5$. Dari teorema master, a = 1, b = 2, d = 0, sehingga $1 = 2^0$ (case 2),
S(m) adalah $O(n^0 \log m) = O(\log m)$. Karena $n = 2^m$, maka $m = {}^2\log n$.
Jadi, T(n) adalah O(log log n)

(c) $T(n) = 2T(\sqrt{n}) + 1$, misalkan $n = 2^m$, maka $T(2^m) = 2T(2^{m/2}) + 1$. Misalkan $S(m) = T(2^m)$,
maka $S(m) = 2S(m/2) + 1$. Dari teorema master, a = 2, b = 2, d = 0, sehingga $2 > 2^0$ (case 3),
S(m) adalah $O(m^{2^{\log 2}}) = O(m)$. Karena $n = 2^m$, maka $m = {}^2\log n$.
Jadi, T(n) adalah O(log n)

Decrease and Conquer

4. Terdapat sebuah larik bilangan bulat [57, 17, 4, 67, 98, 36, 100, 77, 82]. Tanpa melakukan pengurutan larik terlebih dulu, tentukan bilangan terkecil ke-3 dari larik tersebut dengan pendekatan *decrease and conquer by variable size* sesuai salindia kuliah. Catatan: bilangan terkecil ke-1 pada larik tersebut adalah 4, dan bilangan terkecil ke-9 pada larik adalah 100. **(Nilai 15)**

a. Tuliskan tahapan secara global proses pencarian bilangan terkecil ke-3 pada sembarang larik bilangan integer dengan ukuran sembarang dan tidak terurut, menggunakan *decrease and conquer*.
b. Tuliskan dengan lengkap iterasi sesuai jawaban (a) untuk larik pada soal. Satu iterasi adalah hingga partisi dihasilkan.

Jawaban:

a. Gunakan pendekatan quick sort dengan pivot bilangan pertama pada larik masukan, dengan mencari nilai k=3 (terkecil ke-3).
(i) Lakukan partisi larik dengan pendekatan quick sort. Partisi akan menghasilkan setengah bagian yang berisi bilangan lebih kecil atau sama dengan pivot, dan setengah bagian yang berisi bilangan lebih besar atau sama dengan pivot.
(ii) Misal posisi partisi pada indeks s.
Jika s = 3, maka pivot p adalah nilai yang dicari
Jika s > 3, maka bilangan yang dicari terdapat pada setengah bagian kiri
Jika s < 3, maka bilangan yang dicari terdapat pada setengah bagian kanan
(iii) Lanjutkan pencarian dengan kembali ke tahap (i) hanya pada partisi yang berisi bilangan yang dicari.

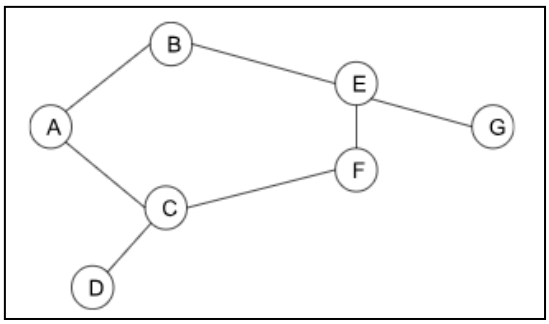
b. Tahapan pencarian pada larik [57, 17, 4, 67, 98, 36, 100, 77, 82].

Iterasi	Keterangan	Indeks dari tiap bilangan 1 2 3 4 5 6 7 8 9
1	larik	57 17 4 67 98 36 100 77 82
	Posisi p/q	pivot p q (tukar isi indeks p dengan isi indeks q)
	larik	57 17 4 36 98 67 100 77 82
	Posisi p/q	q p (p >= q maka dihentikan, tukar posisi pivot)
		36 17 4 57 98 36 100 77 82
		s berada pada indeks = 4, maka s>3; oleh karena itu iterasi berikutnya hanya partisi kiri
2		36 17 4
		pivot q/p (p >= q maka dihentikan, tukar posisi pivot)
		4 17 36
		s berada pada indeks = 3, oleh karena itu bilangan terkecil ke-3 ditemukan.

Jadi bilangan terkecil ke-3 pada larik tersebut adalah 36.

DFS, BFS, IDS

5. Terdapat graf sebagai berikut.



Setiap simpul akan diberi sebuah warna, dan hanya dua warna yang digunakan yaitu merah dan hijau. Pewarnaan dimulai dari simpul A dengan warna merah. Penelusuran (traversal) harus digambarkan sebagai pohon. Sebuah simpul akan diberi warna yang berbeda dengan simpul *parent* pada pohon penelsuran.Jika sebuah simpul memiliki lebih dari satu simpul tetangga, maka yang diperiksa terlebih dahulu sesuai urutan abjad. **(Nilai 20)**

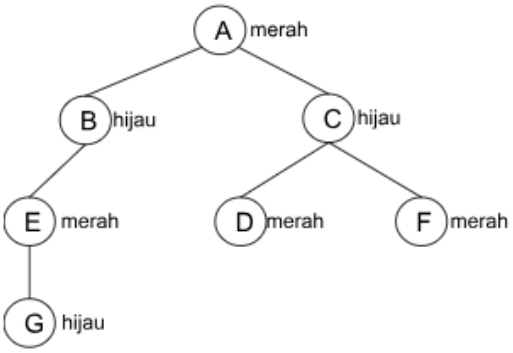
a. Tuliskan urutan pemberian warna simpul jika menggunakan pendekatan **Breadth First Search (BFS),** dengan menggambarkan traversal sebagai pohon BFS**.** Tuliskan warna setiap simpul di sebelah simpul. Contoh:

b. Tuliskan urutan pemberian warna simpul jika menggunakan pendekatan **Depth First Search (DFS),** dengan menggambarkan traversal sebagai pohon DFS**.** Tuliskan warna setiap simpul di sebelah simpul, seperti pada soal (a).

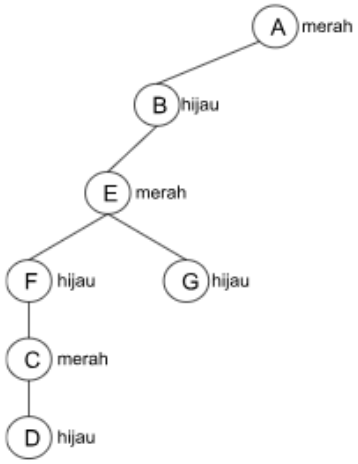
c. Berdasarkan jawaban soal (a) dan (b), apakah graf tersebut bipartit? Jelaskan dengan singkat.

Jawaban:

a. Urutan pemberian warna simpul dengan BFS:

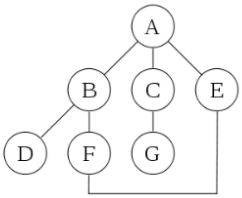


b. Urutan pemberian warna simpul dengan DFS:



c. Graf tersebut bukan graf bipartit, karena ketika dibagi menjadi dua sub graf berdasarkan warna, terdapat sisi yang menghubungkan dua buah simpul dalam sebuah sub graf. Pada Graf bipartit, seharusnya setiap sisi hanya menghubungkan sebuah simpul di sub graf 1 ke simpul di sub graf lain.

6. Diketahui graf di bawah ini :



Tentukanlah bagaimana urutan bekerja dari algoritma IDS (*Iterative Deepening Search*) jika ditelusuri maksimum sampai level kedalaman 3. **(Nilai: 10)**

Jawaban:

Iterasi level 0 : A
Iterasi level 1 : A,B,C,E
Iterasi level 2 : A,B,D,F,C,G, E
Iterasi leve 3 : A,B,D, F,E,C,G,E,F,B

Kapita Selekta

7. Tentukanlah apakah pernyataan di bawah ini salah atau benar, dan kemukakan alasannya :
   a. Strategi algoritmik Divide and Conquer berusaha untuk mencari solusi yang optimum.
   b. Strategi algoritmik Greedy tidak berusaha mencari solusi optimum karena ada kemungkinan untuk terjebak dalam optimum lokal.
   c. Dalam strategi algoritmik IDS (*iterative deepening search*), jika b adalah *branching factor* dan l adalah kedalaman maksimum dari Solusi pertamanya maka space complexity nya adalah $O(b^l)$

(Nilai: 15)

Jawaban:

a. Salah, karena strategi divide and conquer hanya mencari Solusi saja, bukan optimum Solusi.

b. Salah, karena cara kerja strategi algoritma Greedy adalah mencari Solusi optimum, walau memang kadang tidak ketemu Solusi optimumnya karena terjebak dalam optimum local

c. Salah, karena $O(b^l)$ adalah time complexity nya, sedangkan space complexity nya adalah O(bl).