Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

Ujian Akhir Semester IF2211 Strategi Algoritma

Senin, 14 Mei 2018

Waktu: 150 menit

Dosen: Masayu Leylia Khodra, Nur Ulfa Maulidevi, Rinaldi Munir

*Berdoalah terlebih dahulu agar Anda sukses dalam ujian ini!*

**Bagian A (Backtracking dan Route Planning)**

Terdapat sebuah graf sebagai berikut.

2

b

a

2

5

1

4

f

c

2

1

2

3

e

d

Gambar 1. Graf Tak Berarah

1. **(Nilai 15)** Tentukan ‘Hamiltonian Circuit’ dari graf pada Gambar 1, yang diawali pada simpul **a** dan berakhir di simpul **a** dengan menggunakan pendekatan Backtracking. Tentukan:
	1. Representasi Solusi Persoalan (Nilai 3);
	2. Deskripsi Fungsi Pembatas (Nilai 2);
	3. Pohon ruang pencarian lengkap dengan nomor simpul yang menunjukkan urutan pembangkitan. (Nilai 10) Pencarian dihentikan ketika sudah mendapatkan sebuah solusi.

Catatan: prioritas simpul tetangga (atau simpul anak) yang dibangkitkan sesuai urutan abjad, dan ingat perbedaan antara ‘Hamiltonian Circuit’ dan ‘Traveling Salesperson Problem’

1. **(Nilai 15)** Gunakan pendekatan **UCS**, **Greedy Best First** **Search**, dan **A\*** untuk menentukan jalur optimal dari simpul **a** menuju simpul **f**, dengan melengkapi tabel di bawah ini.

|  |  |  |  |
| --- | --- | --- | --- |
| Iterasi | Uniform Cost Search (UCS) | Greedy Best First Search | A \* |
| Formula: f(n) = … {Isikan formula untuk mencari f(n) dengan UCS} | Formula: f(n) = … {Isikan formula untuk mencari f(n) dengan Greedy Best First Search} | Formula: f(n) = … {Isikan formula untuk mencari f(n) dengan A Star} |
| Simpul - Ekspan | Simpul-Hidup | Simpul–Ekspan | Simpul-Hidup | Simpul - Ekspan | SimpulHidup |
| 1 | a | baf(ba) = …caf(ca) = …daf(da) = … | a | baf(ba) = …caf(ca) = …daf(da) = … | a | baf(ba) = …caf(ca) = …daf(da) = … |
|  |
| 2 | … | … |  |  |  |  |
| … |  |  |  |  |  |  |
| Hasil | Jalur = a - … - …Jarak = … | Jalur = a - … - …Jarak = …  | Jalur = a - … - …Jarak = …  |

Catatan: Jika diperlukan heuristik, nilai heuristik sebuah simpul adalah banyaknya busur minimal yang menghubungkan simpul tersebut ke simpul tujuan. Pencarian dihentikan ketika sudah mencapai simpul tujuan. Jika terdapat nilai f(n) terkecil yang sama, maka prioritas simpul yang diperiksa sesuai urutan abjad. Simpul yang sudah diekspan tidak perlu dimasukkan dalam agenda (daftar simpul-hidup).

**Bagian B (Branch&Bound dan Dynamic Programming)**

Untuk Program Dinamis (PD), berikanlah terlebih dahulu fungsi rekurensnya dan bentuklah solusinya per tahap. Solusi *tanpa* tahapan dan fungsi rekurensnya tidak akan diperiksa.

Untuk Branch and Bound (B&B), definisikanlah fungsi pembatas dan nilai setiap simpulnya sesuai fungsi objektifnya, lalu bentuklah pohon ruang status dinamis dengan nomor simpul menyatakan urutan pembangkitan. Nilai untuk akar tetap dihitung. Solusi *tanpa* kedua definisi tersebut tidak akan diperiksa.

1. (**Nilai 20**) Selesaikanlah persoalan Integer Knapsack berikut untuk memaksimumkan profit yang didapatkan dengan Program Dinamis (PD) **dan** Branch and Bound (B&B). Kapasitas maksimum adalah 3.

|  |  |  |
| --- | --- | --- |
| Objek | Bobot | Profit |
| A | 1 | 5 |
| B | 2 | 14 |
| C | 1 | 6 |

1. (**Nilai 15**) Selesaikanlah persoalan TSP berikut untuk meminimumkan jarak sirkuit hamilton yang dilalui dengan Program Dinamis (PD) **atau** Branch and Bound (B&B). Simpul awal = a.



**Bagian C (Pattern Matching dan Teori P dan NP)**

1. **(Nilai 25)**
2. Sebuah string biner sepanjang 100 bit semuanya terdiri dari bit 0. Hitung jumlah perbandingan karakter pada pencocokan *pattern* dengan algoritma *Brute Force* jika *pattern* adalah (i) 0001 dan (ii) 1000.
3. Pertanyaan yang sama dengan (a) jika algoritma yang digunakan adalah Boyer-Moore
4. Diberikan teks “HERE IS A SIMPLE EXAMPLE” dan pola “EXAMPLE”. Gambarkan proses pencocokan pola dengan teks sampai ketemu dengan algoritma KMP dan Boyer-Moore. Hitung jumlah perbandingan karakter yang terjadi. Dalam menjawab soal ini, hitung fungsi pinggiran dan fungsi *last occurence*.

Teks:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| H | E | R | E |  | I | S |  | A |  | S | I | M | P | L | E |  | E | X | A | M | P | L | E |

 Pola:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| E | X | A | M | P | L | E |

1. **(Nilai 10)** Diberikan beberapa buah pernyataan di bawah ini tentang *P*, *NP*, dan *NP-complete*. Tentukan pernyataan mana saja yang benar (sebutkan nomornya saja).
2. *P* Problem adalah himpunan semua persoalan apapun dengan kompleksitas waktu

 polinomial

1. *NP* adalah singkatan dari Non-Polynomial
2. Persoalan di dalam kelas *NP* memiliki waktu polinomial pada tahap verifikasi suatu

 solusi.

1. Sebuah persoalan *X* dikatakan *NP-complete* jika *X* termasuk ke dalam kelas *NP* dan *X* dapat direduksi menjadi persoalan di dalam *NP* lainnya dalam waktu polinomial.
2. Jika *A* adalah sebuah persoalan di dalam *NP-complete* dan *B* adalah persoalan *NP* tapi tidak perlu *NP-complete*, maka jika *A* dapat diselesaikan dalam waktu polinomial maka implikasinya *P* = *NP*.
3. **(Nilai 2)** Apa perkiraan nilai anda untuk mata kuliah ini? (A/AB/B/BC/C/D/E)