
MIDTERM DAA (KOMS119602 / KOMS120403)

Tuesday, March 29th 2022 (07.30 - 10.00 WITA)

Petunjuk

1. Durasi ujian adalah **120 menit** (100 menit pengerjaan soal, dan 20 menit untuk persiapan + submission).
2. Ujian dilaksanakan secara online melalui google meet. Anda diwajibkan terhubung ke google meet selama pelaksanaan ujian.
3. Anda **diharuskan** mengerjakan ujian secara mandiri. Dilarang keras mencontek/berdiskusi dengan siapapun. Ujian ini openbook, namun Anda dilarang mencari materi di Internet.
4. Tulis jawaban dengan singkat (tidak bertele-tele) namun jelas. Tulis dalam Bahasa Indonesia. Hindari menggunakan tinta merah. Kerjakan soal secara berurutan dan tulis semua nomor soal yang dikerjakan meskipun Anda tidak mengerjakannya. Pada setiap lembar jawaban, berikan nomor halaman.

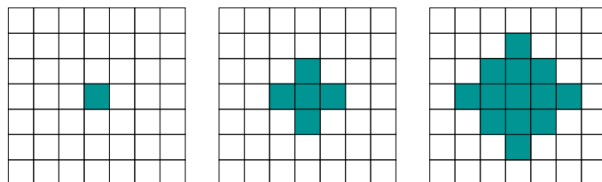
Centang pernyataan berikut pada google form sebelum mengerjakan soal:

“Saya menyatakan bahwa saya mengerjakan UTS ini dengan sejujur-jujurnya, tanpa bantuan orang lain dan tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan UTS ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah DAA Ilkom Semester Genap T.A. 2021/2022.”

1 Algorithms and complexity analysis

1. (Kompleksitas algoritma)

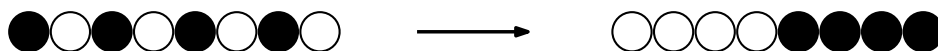
Diberikan algoritma pewarnaan untuk mewarnai kotak satu satuan sebagai berikut: pada iterasi pertama, dimulai dengan mewarnai satu kotak di tengah dan, pada setiap n iterasinya, menambahkan kotak baru di sekelilingnya. Konfigurasi untuk iterasi 1, 2, dan 3 diilustrasikan pada gambar di bawah ini.



- Berapa banyak kotak ukuran satu satuan yang dihasilkan oleh algoritma tersebut pada iterasi ke- i ?
- Nyatakan fungsi kompleksitas waktu algoritma tersebut dengan formula rekursi, kemudian selesaikan formula tersebut untuk mendapatkan fungsi eksplisit kompleksitas waktunya. (*Hint*: Untuk mendapatkan formula rekursif, nyatakan fungsi kompleksitas waktunya dalam suatu formula $f(i) = f(i - 1) + (ai + b)$ untuk suatu konstanta a dan b .)

2. (Sorting)

Diberikan $2n$ bola yang berwarna hitam dan putih masing-masing sebanyak n , dan disusun secara berselang-seling sebagaimana ditunjukkan pada gambar berikut. Anda ditugaskan untuk mengatur susunan bola-bola tersebut sehingga semua bola hitam akan terkumpul di bagian kanan, dan semua bola putih terkumpul di bagian kiri. Pada setiap langkah, Anda hanya diizinkan untuk menukar posisi dua bola yang berurutan.



- Desain sebuah algoritma untuk menyelesaikan masalah ini, dan hitung kompleksitas waktunya (berdasarkan pada jumlah perpindahan yang dilakukan).
- Apakah algoritma yang Anda desain sudah optimal? Berikan argumen singkat, mengapa?

3. (Polynomial interpolation)

Coba ingat kembali algoritma interpolasi polinomial, untuk mengevaluasi polinomial

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$$

di titik $x = t$.

- Algoritma tersebut memiliki kompleksitas waktu *worst case* $\mathcal{O}(n^2)$. Desain sebuah algoritma untuk masalah ini dengan kompleksitas waktu $\mathcal{O}(n)$ (i.e. linier). Jelaskan strateginya dan tuliskan pseudocode untuk algoritma tersebut!
- Apakah mungkin untuk mendesain algoritma untuk masalah ini dengan kompleksitas waktu yang lebih baik dari $\mathcal{O}(n)$? Berikan argumen Anda!

2 Polynomials multiplication

In this exercise, we investigate a divide-and-conquer approach to multiply two polynomials of the same order n (similar to the “Matrix multiplication” and “Large numbers multiplication” discussed in the lecture).

1. (Naive polynomial multiplication algorithm)

Given two polynomials of the same order n as follows. Our goal is to compute $A(x)B(x)$.

$$A(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$$

$$B(x) = b_0 + b_1x + b_2x^2 + \cdots + b_nx^n$$

A naive way to perform the polynomials multiplication is by *direct multiplication*, as in the following example:

Example

$$A(x) = 1 + 2x + 3x^2$$

$$B(x) = 3 + 2x + 2x^2$$

$$A(x)B(x) = (1 + 2x + 3x^2)(3 + 2x + 2x^2) = 3 + 8x + 15x^2 + 10x^3 + 6x^4$$

□

(a) Solve the following polynomials multiplication using naive algorithm!

$$A(x) = 2 + 5x + 3x^2 + x^3 - x^4$$

$$B(x) = 1 + 2x + 2x^2 + 3x^3 + 6x^4$$

(b) Design a brute-force algorithm to multiply two polynomials $A(x)$ and $B(x)$ that are of order n .

(c) Compute the complexity of your brute-force algorithm. Represent it using an asymptotic notation!

2. (Polynomials multiplication divide-and-conquer algorithm)

Given two polynomials of the same order n as follows. Our goal is to compute $A(x)B(x)$.

$$A(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$$

$$B(x) = b_0 + b_1x + b_2x^2 + \cdots + b_nx^n$$

How do we perform polynomials multiplication by Divide-and-Conquer? The algorithm is as follows.

- Split $A(x)$ into $A_0(x)$ and $A_1(x)$, each contains $n/2$ terms:

$$A_0(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{\lceil n/2 \rceil - 1}x^{\lceil n/2 \rceil - 1}$$

$$A_1(x) = a_{\lceil n/2 \rceil} + a_{\lceil n/2 \rceil + 1}x + a_{\lceil n/2 \rceil + 2}x^2 + \cdots + a_{n - \lceil n/2 \rceil}x^{n - \lceil n/2 \rceil}$$

So that

$$A(x) = A_0(x) + A_1(x)x^{\lceil n/2 \rceil}$$

- Similarly, $B(x)$ can be split into $B_0(x)$ and $B_1(x)$, so that:

$$B(x) = B_0(x) + B_1(x)x^{\lceil n/2 \rceil}$$

Hence:

$$A(x)B(x) = A_0(x)B_0(x) + (A_0(x)B_1(x) + A_1(x)B_0(x))x^{\lceil n/2 \rceil} + A_1(x)B_1(x)x^{2\lceil n/2 \rceil}$$

Task: Solve the following polynomials multiplication using the algorithm explained above. Write the steps clearly!

$$A(x) = 2 + 5x + 3x^2 + x^3 - x^4$$

$$B(x) = 1 + 2x + 2x^2 + 3x^3 + 6x^4$$

3. (DnC-based polynomials multiplication: pseudocode and time complexity)

The divide-and-conquer algorithm explained in question 3 can be written in a pseudocode as follows:

Algorithm 3 Polynomials multiplication (divide-and-conquer, version 1)

```

1: procedure POLYMUL( $A, B$ : polynomials,  $n$ : integer)
2:   declaration
3:      $A_0, A_1, B_0, B_1$ : polynomials
4:      $s$ : integer
5:   end declaration
6:   if  $n = 0$  then return  $A * B$  ▷ scalar multiplication
7:   else
8:      $s \leftarrow \lceil n/2 \rceil$ 
9:      $A_0 \leftarrow a_0 + a_1x + a_2x^2 + \dots + a_{s-1}x^{s-1}$ 
10:     $A_1 \leftarrow a_sx^s + a_{s+1}x^{s+1} + a_{s+2}x^{s+2} + \dots + a_nx^{n-s}$ 
11:     $B_0 \leftarrow b_0 + b_1x + b_2x^2 + \dots + b_{s-1}x^{s-1}$ 
12:     $B_1 \leftarrow b_sx^s + b_{s+1}x^{s+1} + b_{s+2}x^{s+2} + \dots + b_nx^{n-s}$ 
13:    return  $\text{POLYMUL}(A_0, B_0, s) + \text{POLYMUL}(A_0, B_1, s) + \text{POLYMUL}(A_1, B_0, s) * x^s + \text{POLYMUL}(A_1, B_1, s) * x^{2s}$ 
14:   end if
15: end procedure

```

Task: Write the time complexity function of the divide-and-conquer algorithm above in a recursive formula. Using Master theorem, compute the asymptotic time complexity!

4. (DnC-based polynomials multiplication: improvement)

Now we want to modify the divide-and-conquer algorithm for the polynomials multiplication given in question 2. We will reduce the number of multiplications performed in the algorithm.

In question 2, we have:

$$A(x)B(x) = A_0(x)B_0(x) + (A_0(x)B_1(x) + A_1(x)B_0(x))x^{\lceil n/2 \rceil} + A_1(x)B_1(x)x^{2\lceil n/2 \rceil}$$

There are 4 multiplications and 3 additions of polynomials of order n . We will reduce the number of multiplications to 3, but with a consequence that the number of additions is increased.

Define:

$$Y(x) = (A_0(x) + A_1(x)) \times (B_0(x) + B_1(x))$$

$$U(x) = A_0(x)B_0(x)$$

$$Z(x) = A_1(x)B_1(x)$$

Then:

$$Y(x) - U(x) - Z(x) = A_0(x)B_1(x) + A_1(x)B_0(x)$$

so that:

$$\begin{aligned} A(x)B(x) &= A_0(x)B_0(x) + (A_0(x)B_1(x) + A_1(x)B_0(x))x^{\lceil n/2 \rceil} + A_1(x)B_1(x)x^{2\lceil n/2 \rceil} \\ &= U(x) + (Y(x) - U(x) - Z(x))x^{\lceil n/2 \rceil} + Z(x)x^{2\lceil n/2 \rceil} \end{aligned}$$

Note that in this algorithm, there are only three multiplications, namely $U(x)$, $Y(x)$, and $Z(x)$.

Task: Solve the following polynomials multiplication using the algorithm explained above. Write the steps clearly!

$$\begin{aligned} A(x) &= 2 + 5x + 3x^2 + x^3 - x^4 \\ B(x) &= 1 + 2x + 2x^2 + 3x^3 + 6x^4 \end{aligned}$$

5. (DnC poly-multiplication improvement: pseudocode)

The divide-and-conquer algorithm explained in question 3 can be written in a pseudocode as follows:

Algorithm 4 Polynomials multiplication (divide-and-conquer, version 2)

```

1: procedure POLYMUL2( $A, B$ : polynomials,  $n$ : integer)
2:   declaration
3:      $A_0, A_1, B_0, B_1, U, Y, Z$ : polynomials
4:      $s$ : integer
5:   end declaration
6:   if  $n = 0$  then
7:     return  $A * B$  ▷ scalar multiplication
8:   else
9:      $s \leftarrow \lceil n/2 \rceil$ 
10:     $A_0 \leftarrow a_0 + a_1x + a_2x^2 + \dots + a_{s-1}x^{s-1}$ 
11:     $A_1 \leftarrow a_sx^s + a_{s+1}x^{s+1} + a_{s+2}x^{s+2} + \dots + a_nx^{n-s}$ 
12:     $B_0 \leftarrow b_0 + b_1x + b_2x^2 + \dots + b_{s-1}x^{s-1}$ 
13:     $B_1 \leftarrow b_sx^s + b_{s+1}x^{s+1} + b_{s+2}x^{s+2} + \dots + b_nx^{n-s}$ 
14:     $Y \leftarrow \text{POLYMUL2}(A_0 + A_1, B_0 + B_1, s)$ 
15:     $U \leftarrow \text{POLYMUL2}(A_0, B_0, s)$ 
16:     $Z \leftarrow \text{POLYMUL2}(A_1, B_1, s)$ 
17:    return  $U + (Y - U - Z) * x^s + Z * x^{2s}$ 
18:   end if
19: end procedure

```

Task: Write the time complexity function of the divide-and-conquer algorithm above in a recursive formula. Using Master Theorem, compute the asymptotic time complexity! What do you observe?

6. (Analysis)

After computing the result of $A(x)B(x)$, check if the three different algorithms (brute-force, divide-and-conquer, and the improved version of divide-and-conquer) above give the same result. If not, explain why!