Program Dinamis dalam Memaksimalkan Keuntungan Pemotongan Batangan Baja Pada Pabrik Konstruksi

Restu Wahyu Kartiko - 13516155[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]rwk03@itb.ac.id

***Abstract*— Dalam proses konstruksi, ada empat komponen utama yaitu pasir, semen, batu, dan juga baja. Setiap proses konstruksi membutuhkan jumlah dari keempat elemen diatas berbeda-beda. Ukuran baja yang digunakan juga berbeda-beda sesuai kebutuhan konstruksi. Dari sudut pandang pabrik produsen baja, pemotongan batangan baja memerlukan konfirgurasi ukuran pemotongan yang paling optimal agar menghasilkan potongan-potongan baja yang mempunyai profit yang paling besar. Makalah kali ini akan membahas tentang strategi pemilihan konfigurasi potongan baja sesuai kebutuhan pasar agar menghasilkan keuntungan yang lebih besar.**

***Keywords*— Program dinamis, greedy, batangan baja, konstruksi.**

I. PENDAHULUAN

Indonesia merupakan salah satu negara yang sedang menyemarakkan masalah pembangunan guna mengimbangi proses modernisasi global yang sedang terjadi. Perpindahan penduduk dari desa kekota terjadi di kota-kota di Indonesia terutama di kota besar seperti Jakarta, Bandung, dan Surabaya. Untuk mengimbangi pertambahan penduduk tersebut, banyak dilakukan pembangunan infrastruktur terutama gedung-gedung baik itu gedung perkantoran, perbelanjaan, atau gedung tempat tinggal.

Pembangunan infrastruktur dilakukan oleh pihak pemerintah maupun pihak swasta. Pihak pemerintah biasanya membangun infrastruktur didaerah secara menyebar dan luas, sementar pihak swasta biasanya membangun infrastruktur secara terpusat seperti BSD city, Meikarta, dll.

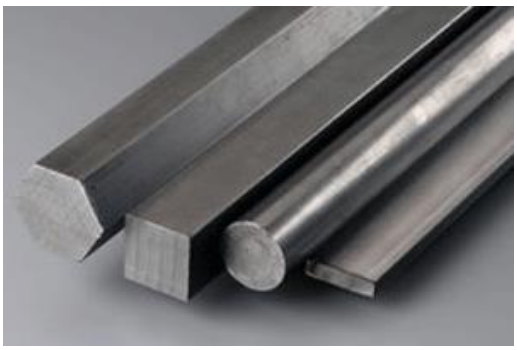
Dalam proses pembangunan infrastruktur, pengembang akan menjalin kontrak dengan kontraktor. Selaku pelaku pembangunan, kontraktor memerlukan ukuran baja yang beragam. Hal ini menimbulkan dampak perbedaan pada harga baja sesuai dengan panjang baja tersebut.

Baja batangan adalah salah satu bahan utama dalam proses pembangunan selain semen, batu, dan pasir. Kontrktor biasa mendapatkan baja batangan dari pabrik konstruksi baja. Dari sisi pabrik konstruksi baja, hal ini bisa dimanfaatkan untuk menentukan strategi pemotongan ukuran baja agar hasil penjualan mereka memiliki keuntungan paling besar sehingga keuntungan perusahaan bertambah.

II. TEORI DASAR

A. Batangan Baja

Batangan baja merupakan salah satu bahan dasar bangunan yang memiliki peranan penting dalam berdirinya suatu bangunan. Batangan baja menjadi fondasi dari berdirinya bangunan karena dipakai hampir diseluruh bagian bangunan sehingga pemilihan batangan baja yang bermutu akan mempengaruhi kekuatan suatu bangunan.



Ukuran batangan baja yang dibutuhkan pada sebuah konstruksi bangunan berbeda-beda. Oleh karena itu, jumlah permintaan akan batangan baja ukuran N dengan batangan baja ukuran M akan berbeda. Menurut prinsip ekonomi, semakin besar permintaan, semakin besar pula harga jual. Perbedaan harga jual ini lah yang mendasari pemikiran pabrik untuk memilih konfigurasi pemotongan yang sesuai agar total harga jual menjadi maksimum.

Berikut akan diberikan contoh kasus. Misalnya suatu pabrik baja mempunyai baja-baja yang berukuran 80 meter. Saat melihat permintaan pasar, ternyata harga baja adalah sebagai berikut:

Panjang :	10	20	30	40	50	60	70	80
Harga : (dalam juta)	10	50	80	90	100	170	170	220

Dari data diatas, maka pemotongan optimum tiap baja yang berukuran 80 meter adalah menjadi baja yang berukuran 20 meter dan 60 meter yang akan memberikan total harga 220 juta.

B. Algoritma Greedy

Algoritma Greedy adalah algoritma yang paling populer untuk memecahkan persoalan optimasi. Greedy sendiri diambil dari bahasa inggris yang artinya rakus, tamak atau serakah .Prinsip algoritma greedy adalah: "take what you can get now!"[1]. Algoritma greedy mengambil keputusan yang paling optimal pada setiap langkah. Maka dari itu algoritma ini membentuk solusi langkah per langkah. Keputusan yang sudah diambil pada suatu langkah tidak bisa diulangi pada langkah selanjutnya. Karena Greedy mengambil keputusan paling optimal disetiap langkah, maka seringkali solusi yang diberikan bersifat optimum local, bukan optimum global.

Persoalan optimasi dalam konteks algoritma greedy disusun oleh elemen-elemen sebagai berikut[2]:

1. Himpunan kandidat, C.
   Himpunan ini berisi elemen-elemen pembentuk solusi. Contohnya adalah himpunan koin, himpunan pekerjaan yang akan dikerjakan, himpunan harga baja, dan lain-lain. Pada setiap langkah, satu buah kandidat diambil dari himpunan tersebut.
2. Himpunan solusi, S.
   Berisi kandidat-kandidat yang terpilih sebagai solusi persoalan. Himpunan solusi adalah himpunan bagian dari himpunan kandidat yang akan membetuk solusi dari persoalan.
3. Fungsi seleksi
   Fungsi seleksi adalah fungsi yang pada setiap langkah memilih kandidat yang akan mencapai solusi optimal. Kandidat yang sudah dipilih pada suatu langkah tidak pernah dipertimbangkan lagi pada langkah selanjutnya.
4. Fungsi kelayakan
   Fungsi kelayakan adalah fungsi yang memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yakni kandidat tersebut bersama-sama dengan himpunan solusi yang sudah terbentuk tidak melanggar kendala (constrains) yang ada. Kandidat yang layak dimasukkan ke dalam himpunan solusi, sementara yang tidak layak dibuang dan tidak pernah dipertimbangkan lagi.
5. Fungsi obyektif
   Fungsi obyektif merupakan fungsi yang memaksimumkan atau meminimumkan nilai solusi tergantung jenis masalah (maximization atau minimization).

Skema umum algoritma greedy:

```
function greedy(input C: himpunan_kandidat)
→ himpunan_kandidat
{
Mengembalikan solusi dari persoalan optimasi
dengan algoritma greedy
Masukan : himpunan kandidat C
Keluaran : himpunan solusi yang bertipe
himpunan kandidat
}

Deklarasi
   x : kandidat
   S : himpunan_kandidat

Algoritma:
   S ← {}    { inisialisasi S dengan kosong
   }
   while (not SOLUSI(S)) and (C ≠ {} ) do
       x ← SELEKSI(C)  { pilih sebuah
   kandidat dari C}
       C ← C - {x}  {  elemen  himpunan
   kandidat berkurang satu }
       if LAYAK(S ∪ {x}) then
          S ← S ∪ {x}
       endif
   endwhile {SOLUSI(S) or C = {} }

   if SOLUSI(S) then
         return S
   else
       write(' tidak ada solusi' )
endif
```

C. Program Dinamis

Program dinamis adalah metode pemecahan masalah dengan pendekatan optimisasi yang mengubah masalah kompleks menjadi uruaian masalah yang lebih sederhana. karakteristik utamanya adalah sifat multistage dari prosedur optimasi. Solusi dari persoalan dapat dipandang dari serangkaian keputusan yang saling berkaitan.

Perbedaan utama antara algoritma greedy dengan program dinamis adalah greedy hanya menghasilkan satu rangkaian keputusan, sementara program dinamis menghasilkan lebih dari satu rangkaian keputusan[3].

Berikut ini merupakan karakteristik persoalan yang dapat diselesaikan dengan metode program dinamis[3].

1. Persoalan dapat dibagi menjadi beberapa tahap yang pada setiap tahap hanya diambil satu keputusan.
2. Masing-masing tahap terdiri dari sejumlah status yang berhubungan dengan tahap tersebut. Secara umum status merupakan bermacam kemungkinan masukan yang ada pada tahap tersebut. Jumlah status bisa berhingga maupun tak hingga.
3. Hasil dari keputusan yang diambil pada setiap tahap ditrasnformasikan dari status yang bersangkutan ke status berikutnya pada tahap berikutnya.
4. Ongkos pada suatu tahap meningkat secara teratur dengan bertambahnya jumlah tahapan.
5. Ongkos pada suatu tahap bergantung pada ongkos tahap-taha yang sudah berjalan dan ongkos pada tahap tersebut.
6. Keputusan terbaik pada suatu tahap bersifat independen terhadap keputusan yang dilakukan pada tahap sebelumnya
7. Adanya hubungan rekursif yang mengidentifikasikan keputusan terbaik untuk setiap status pada tahap k memberikan keputusan terbaik pada setiap status pada tahap k+1
8. Prinsip optimaalitas berlaku pada persoalan tersebut

Secara umum ada empat langkah yang dilakukan dalam mengembangkan algortima program dinamis[3]:

1. Karakteristikkan struktur solusi optimal
2. Definisikan secara rekursif nilai solusi optimal
3. Hitung nilai solusi optimal secara maju atau mundur
4. Konstruksikan solusi optimal

Ada dua pendekatan yang digunakan dalam membuat program dinamis yaitu program dinamis maju dan program Program dinamis maju bergerak mulai dari tahap 1, terus maju ke tahap 2, 3, dan seterusnya sampai tahap n. Program dinamis mundur bergerak mulai dari tahap n, terus mundur ke tahap n – 1, n – 2, dan seterusnya sampai tahap 1.

III. PEMBAHASAN

Pada pembahasan kali ini, akan digunakan asumsi seperti pada point A ditambah dengan asumsi bahwa pada pabrik baja tersebut, hanya memproduksi satu jenis ukuran batangan baja standart (dalam kasus diatas berukuran 80 meter). Dalam pembahasan ini, kita akan mencoba mengevaluasi keuntungan masksimum yang bisa kita hasilkan dari potongan baja tersebut menggunakan algoritma greedy dan program dinamis.

A. Pemodelan Dengan Algoritma Greedy

Pada persoalan ini, ditentukan elemen-elemen algortima greedynya sebagai berikut:

1. Himpunan kandidat
   Himpunan harga pasar batangan besi pada ukuran 1 hingga n.
2. Himpunan solusi
   Himpunan panjang batangan besi yang terpilih.
3. Fungsi seleksi
   Fungsi ini akan memilih batangan besi yang memiliki nilai perbandingan harga/panjang yang paling tingga.
4. Fungsi layak
   Fungsi layak disini adalah memeriksa apakah nilai total panjang yang ada di himpunan solusi tidak melebihi panjang batangan besi standars.
5. Fungsi objektif
   Fungsi objektifnya adalah untuk mendapatkan jumlah harga yang paling besar dari batangan besi tersebut.

Berikut adalah kode yang telah penulis buat untuk algoritma greedy

```
while(n){
    maksimal = 0;
    for(int i = 1;i <= n;i++){
        if( (n-i) >= 0 ){
            maksimal = price[i];
            indeks = i;
        }
    }
    n -= indeks;
    answer += maksimal;
}

return answer;
```

Agar lebih jelas, akan diberikan contoh penyelesaian kasus dengan algoritma greedy dengan panjang batangan besi standart adalah 80m:

Panjang(m) :	1	2	3	4	5
Harga (dalam juta) :	1	5	7	10	11
Harga/Panjang :	1	2.5	2.33	2.5	2.2

Pada kasus diatas dengan n = 5, himpunan solusi S = {}. Berikut merupakan langkah yang diambil tiap iterasi:

1. N = 5
   Harga/panjang yang terbesar dan panjang < N adalah panjang = 3. Masukkan 3 kehimpunan Solusi.
   S = {3}
   N = 5 – 4 = 1.
2. N = 1
   Harga/panjang yang terbesar dan panjang < N adalah panjang = 1. Masukkan 1 kehimpunan Solusi.
   S = {7, 1}
   N = 1 – 1 = 0

Hasil dari algoritama greedy setelah dijalankan menemukan pemotongan paling optimal adalah S = {4,1}, sehingga total

harga yang bisa didapat adalah harga[4] + harga[1] = 10 + 1 = 11 juta tiap batangan besi standart.

Algoritma greedy disini hanya digunakan sebagai pembanding dengan program dinamis.

B. Pemodelan Dengan Program Dinamis

Permasalahan ini sesuai dengan kriteria program dinamis karena ada beberapa tahapan yang akan dilakukan berulang ulang. Misalnya dengan panjang 8, kita telah memilih potongan besi dengan panjang 3, maka sekarang kita memiliki potongan besi dengan panjang 3 dan 6 meter. Di kasus lain, jika kita memilih potongan dengan panjang 2 dan 1 dari besi standart dengan panjang 8, maka kita memiliki potongan dengan panjang 2, 1, dan 6 meter. Dapat kita lihat bahwa kedua cabang kasus diatas akan menghitung nilai optimal saat potongan besi memiliki panjan 6 meter. Hal itu dapat kita optimasi dengan menggunakan program dinamis sehingga kita tidak perlu mencari nilai optimum untuk memotong 6 meter berkali-kali.

Program dinamis akan mencari harga maksimum pada setiap tahap dimana tahap adalah nilai I < N, I >= 0 dan I adalah panjang batangan besia.

Berikut akan dijelaskan penyelesaian kasus menggunakan Program dinamis.

1. Struktur Data
   Struktur data yang digunakan adalah array dengan indeks maksimum N. Array ini akan menyimpan harga maksimum yang bisa kita peroleh jika kita memiliki panjang batangan baja I dimana I adalah indeks dari array ini. Definisi formalnya, array[I] adalah nilai harga masksimum saat batangan baja memiliki panjang I.
2. Program Utama
   Dalam kasus ini, akan digunakan program dinamis maju. Nilai maksimum pada tiap tahap dapat dinyatakan sebagai berikut:

   DP(0) = 0 // basis
   DP(N) = max(harga[j] + DP(i-j)
   untuk setiap j <= N dan j >= 1)

   Secara detail, pada setiap tahap i, program dinamis akan melakukan beberapa hal sebagai berikut:
   a. Inisiasi nilai semua F[] dengan nilai yang sangat kecil.
   b. Inisiasi nilai i dengan 1.
   c. Lakukan iterasi dari 1 hingga ke i. Pada tiap iterasi j, lakukan assignment berikut

      DP(i) = max (DP(i), harga[j] + DP(i-j))

      Array penyimpanan DP(i) akan dipakai pada tahap selanjutnya
   d. Tambahkan i dengan 1.
   e. Jika i kurang dari sama dengan N, kembali ke langkah c.
   f. Nilai DP(N) merupakan harga maksimum yang dapat kita peroleh.

Berikut kode program yang sudah penulis buat:

```
int ProgramDinamis(double price[], int n)
{
   double DP[n+1];
   int i, j;

   for (i = 1; i <= n; i++)
   {
       DP[i] = -999999.0; // inisiasi array DP
       for (j = 1; j <= i; j++)
         DP[i] = max(DP[i], price[j] + DP[i-j]);
   }

  return DP[n];
}
```

Supaya lebih jelas, akan dijelaskan dengan ilustrasi berikut:

Misalkan ada data harga tiap panjang potongan baja sebagai berikut seperti pada contoh sebelumnya

Panjang(m) :	1	2	3	4	5
Harga (dalam juta) :	1	5	7	10	11

Diawal penyelesaian, akan dibuat array dengan indeks 1 hingga N dan assign seluruhnya dengan nilai yang sangat keci. Misal -∞. Array tersebut akan berisi seperti berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	-∞	-∞	-∞	-∞	-∞

Kita akan melakukan iterasi i mulai dari 1 hingga N.

Saat i = 1 dan j = 1

DP(1) = max (DP(1), harga[1] + DP(0))
= max (-∞, 1 + 0)
DP(1) = 1

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	-∞	-∞	-∞	-∞

Saat i = 2 dan j = 1

DP(2) = max (DP(2), harga[1] + DP(1))
= max (-∞, 1 + 1)
DP(2) = 2

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	2	-∞	-∞	-∞

Saat i = 2 dan j = 2

DP(2) = max (DP(2), harga[2] + DP(0))
= max (2, 5 + 0)
DP(2) = 5

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	-∞	-∞	-∞

Saat i = 3 dan j = 1

DP(3) = max (DP(3), harga[1] + DP(2))
= max (-∞, 1 + 5)
DP(3) = 6

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	6	-∞	-∞

Saat i = 3 dan j = 2

DP(3) = max (DP(3), harga[2] + DP(1))
= max (6, 5 + 1)
DP(3) = 6

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	6	-∞	-∞

Saat i = 3 dan j = 3

DP(3) = max (DP(3), harga[3] + DP(0))
= max(6, 7 + 0)
DP(3) = 7

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	7	-∞	-∞

Saat i = 4 dan j = 1

DP(4) = max (DP(4), harga[1] + DP(3))
= max(-∞, 1 + 7)
DP(4) = 8

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	7	8	-∞

Saat i = 4 dan j = 2

DP(4) = max (DP(4), harga[2] + DP(2))
= max (8, 5 + 5)
DP(4) = 10

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	7	10	-∞

Saat i = 4 dan j = 3

DP(4) = max (DP(4), harga[3] + DP(1))
= max (10, 7 + 1)
DP(4) = 10

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	7	10	-∞

Saat i = 4 dan j = 4

DP(4) = max (DP(4), harga[4] + DP(0))
= max (10, 10 + 0)
DP(4) = 10

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	7	10	-∞

Saat i = 5 dan j = 1

DP(5) = max (DP(5), harga[1] + DP(4))
= max (-∞, 1 + 10)
DP(5)= 11

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	7	10	11

Saat i = 5 dan j = 2

DP(5) = max (DP(5), harga[2] + DP(3))
= max (11, 5 + 6)
DP(5) = 11

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	5	7	10	11

Saat i = 5 dan j = 3

DP(5) = max (DP(5), harga[3] + DP(2))
= max (11, 7 + 5)
DP(5) = 12

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	4	6	10	12

Saat i = 5 dan j = 4

DP(5) = max (DP(5), harga[4] + DP(1))
= max (12, 10 + 1)
DP(5) = 12

Setalah melakukan perbandingan tersebut, maka

array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	4	6	10	12

Saat i = 5 dan j = 5

DP(5) = max (DP(5), harga[5] + DP(0))
= max (12, 11 + 0)
DP(5) = 12

Setalah melakukan perbandingan tersebut, maka array akan terlihat sebagai berikut,

Indeks DP	0	1	2	3	4	5
Nilai DP	0	1	4	6	10	12

Berikut contoh keluaran program yang penulis buat

```
Masukkan panjang bantangan baja standart : 5
Harga baja dengan panjang 1 : 1
Harga baja dengan panjang 2 : 5
Harga baja dengan panjang 3 : 7
Harga baja dengan panjang 4 : 10
Harga baja dengan panjang 5 : 11
Harga maksimal yang bisa diperoleh adalah : 12 juta
```

Nilai dari DP[5] adalah harga maksimum yang dapat kita peroleh dari pemotongan besi standart dengan panjang 5, hargnya adalah 12 juta.

C. Perbandingan Hasil Algoritma Greedy Dengan Program Dinamis

Dari hasil pembahasan diatas, penulis telah melakukan uji coba dengan test case yang sama, algoritma greedy menghasilkan nilai 11 juta. Sementara program dinamis menghasilkan nilai 12 juta. Hal ini membuktikan bahwa program dinamis menghasilkan hasil yang paling optimal.

Dari segi komplesitas waktu, algortima greedy memiliki t(n) = n. sementar algoritma program dinamis memiliki t(n) = n*(n+1)/2. Walaupun greedy relatif lebih cepat, namun program dinamis tidak terlalu terpaut jauh dan memberikan solusi optimal.

IV. KESIMPULAN

Program dinamis memiliki banyak penerapan pada dunia nyata salah satunya pada penentual konfigurasi pemotongan batangan baja.

Konfigurasi pemotongan yang efektif disertai pengetahuan tentang harga pasar akan memberikan keuntungan yang paling besar bagi perusahaan. Konfigurasi pemotongan dapat dicari dengan berbagai macam algoritma salah satunya dengan greedy dan program dinamis.

Dari hasil diatas, program dinamis mampu memberikan solusi yang lebih optimal dan selalu memberikan solusi optimal karena program dinamis menghitung semua konfigurasi pemotongan yang ada dan menyimpang konfigurasi pada suatu tahap agar tidak perlu menghitung kembali jika program membangkitkna tahap tersebut.

V. UCAPAN TERIMA KASIH

Penulis mengucapkan terimakasih kepada Allah SWT atas berkat dan rahmat-Nya, penulis dapat menyelsaikan makalah ini dengan sebaik mungkin. Penulis juga ingin mengucapkan terima kasih kepada kedua orang tua penulis yang telah memberikan dukungan kepada penulis dalam menjalankan perkuliahan. Selain itu penulis mengucapkan terimakasih kepada Ibu Nur Ulfa Maulidevi selaku dosen mata kuliah Strategi Algoritma K-02 untuk semua ilmu yang mereka berikan yang sangat berguna dalam pembuatan makalah ini. Terkahir penulis ingin mengucapkan kepada semua pihak yang telah memberi dukungan dan inspirasi dalam membuat makalah ini.

REFERENCES

[1] https://www.it-jurnal.com/pengertian-algoritma-greedy/
[2] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Greedy-(2018).pdf
[3] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Program-Dinamis-(2018).pdf

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 13 Desember 2018

Restu Wahyu Kartiko, 13516155