Perbandingan Algoritma *Knuth-Morris-Pratt* dan Algoritma *Boyer-Moore* dalam Pencarian Teks di Bahasa Indonesia dan Inggris

Kevin Wibowo-13509065
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13509065@std.stei.itb.ac.id

***Abstract*—Makalah ini membahas tentang perbandingan kecepatan dan efektivitas algoritma Knuth-Morris-Pratt(KMP) dengan algoritma Boyer-Moore yang merupakan algoritma pencarian string yang umum digunakan. Makalah ini akan membandingkan dengan tujuan mencari algoritma yang lebih efektif dalam penenrapannya di kehidupan sehari-hari. Perbandingan akan dilakukan dengan melakukan pengujian menggunakan artikel dari berbagai Bahasa seperti Indonesia, dan Inggris. Teknik pencarian yang digunakan untuk mencari dari kemunculan sebuah substring.**

***Index Terms*—Knuth-Morris-Pratt, Boyer-Moore, String**

I. PENDAHULUAN

Dalam kehidupan sehari-hari penggunaan berbagai macam aplikasi dan program komputer telah berkembang pesat. Salah satu bentuk fitur yang paling sering digunakan dan terdapat dalam berbagai aplikasi adalah algoritma pencarian *string*, dimana pengguna dapat mencari kata, frasa, atau kalimat untuk mempermudah pengaksesan dan pencarian dalam berbagai hal, seperti dalam pencarian di suatu dokumen atau *e-book*, pencarian suatu file atau program di komputer, bahakan pencarian menggunakan *search engine* di internet telah menjadi sangat populer.

Oleh karena itu diperlukan adanya pemahaman lebih lanjut tentang pemBahasan mengenai kedua algoritma tersebut, apabila pengembangan fitur *string searching* akan dilakukan lebih lanjut. Dalam makalah ini akan dibahas mengenai perbandingan antara kedua algoritma pencarian *string* yaitu algoritma *Knuth-Morris-Pratt*, dan algoritma *Boyer-Moore* untuk mencari mana yang lebih efektif dipakai dalam kehidupan sehari-hari.

II. DASAR TEORI

2.1 Algoritma Pencarian *String*

Algoritma pencarian *string* adalah salah satu jenis algoritma yang mencari sebuah pola dalam suatu pola yang lebih besar. Pola tersebut dapat berupa kalimat, artikel atau bahkan gambar. Terdapat berbagai jenis algoritma yang dapat digunakan seperti *brute force*, *Knuth-Morris-Pratt, Boyer-Moore, Rabin-Karp, Aho-Corasick*, dan lain-lain. Penerapan algoritma pencarian pola ini dapat ditemukan dalam berbagai bidang seperti biologi dalam pencarian untaian DNA, informatika dalam *search engine*, dan pendeteksian plagiarism dalam karya tulis.

2.2 Algoritma *Knuth-Morris-Pratt*

Algoritma *Knuth-Morris-Pratt* adalah algoritma pencarian *string* yang mencari dengan cara menghitung dari dimulai dari ketidakcocokan ditemukan, dari ketidakcoockan tersebut akan dhitung dari mana pencarian selanjutnya sebaiknya dimulai. Algoritma *Knuth-Morris-Pratt* menggunakan fungsi pembatas (*border function)* yang digunakan untuk menghitung urutan keberapa perbandingan harus dilakukan. *Border function* dihitung dengan menghitung *panjang prefix* yang ada disebuah *pattern* yang sama dengan *suffix*-nya.

Contoh Algoritma KMP dalam Java:

```
  public static int kmpMatch(String text, String pattern)
{
  int n = text.length();
  int m = pattern.length();
  int fail[] = computeFail(pattern);
  int i=0;
  int j=0;
    while (i < n) {
   if (pattern.charAt(j) == text.charAt(i)) {
     if (j == m - 1)
       return i - m + 1; // match
     i++;
     j++;
   }
   else if (j > 0)
     j = fail[j-1];
   else
     i++;
  }
  return -1; // no match
}
   public static int[] computeFail(String pattern)
{
  int fail[] = new int[pattern.length()];
```

```
  fail[0] = 0;

  int m = pattern.length();
  int j = 0;
  int i = 1;
      while (i < m) {
   if  (pattern.charAt(j) ==
       pattern.charAt(i)) {   //j+1 chars match
    fail[i] = j + 1;
    i++;
    j++;
   }
   else if (j > 0) // j follows matching prefix
    j = fail[j-1];
   else {     // no match
    fail[i] = 0;
    i++;
   }
  }
  return fail;
 }
```

2.3 Algoritma *Boyer-Moore*

Algoritma *Boyer-Moore* adalah algoritma pencarian *string* yang mencari dengan cara membandingkan sebuah huruf dengan huruf yang ada di *pattern* yang dicari, dan menggeser *pattern* tersebut hingga posisinya sama dengan teks yang dicari dan membandingkan kata tersebut. Cara ini disebut *character jump.*

Contoh Algoritma *Boyer-Moore* dalam *Java:*

```
public static int bmMatch(String text, String pattern)
 {int last[] = buildLast(pattern);
  int n = text.length();
  int m = pattern.length();
  int i = m-1;
  if (i > n-1)
   return -1;
  int j = m-1;
  do {
   if (pattern.charAt(j) == text.charAt(i))
    if (j == 0)
     return i; // match
    else {         i--;
     j--;
    }
   else { int lo = last[text.charAt(i)];
     i = i + m - Math.min(j, 1+lo);
     j = m - 1;
   }
  } while (i <= n-1);
  return -1;
 }
 public static int[] buildLast(String pattern)
{
  int last[] = new int[128]; // ASCII char set

  for(int i=0; i < 128; i++)
   last[i] = -1; // initialize array

  for (int i = 0; i < pattern.length(); i++)
   last[pattern.charAt(i)] = i;

  return last;
 }
```

III. DATA PENGETESAN

Pengetesan yang dilakukan akan menggunakan algoritma yang ditampilkan sebgai contoh diatas dalam Bahasa Java dan juga menggunkan *library* dari dari *System* yang digunkan untuk menghitung waktu.

3.1 Penggunaan dalam Bahasa Inggris

Untuk pengetesan dalam Bahasa Inggris akan digunakan beberapa artikel yang diambil dari internet:
Artikel 1 diambil dari:
http://www.upi.com/Business_News/Security-Industry/2011/12/07/Smartphone-probe-exposes-computer-software-and-hardware-risks/UPI-25881323292066/?spt=hs&or=si

Artikel tersebut memiliki panjang 613 kata dan 4373 karakter dengan kata yang dicari adalah instances yang merupakan kalimat ke 602 dari artikel tersebut. Dengan melakukan 10 kali Pengetesan masing-masing dengan menggunakan *KMP* dan *Boyer-Moore*. Hasil yang dihasilkan dapat dilihat di tabel sebagai berikut.

No.	KMP	BM
1	461022	187664
2	426801	174114
3	422610	175581
4	485606	182426
5	418279	185080
6	541549	177467
7	421143	178444
8	418559	175232
9	419606	176699
10	419537	173695

Tabel 3.1 Pengetesan 1: Hasil Pengetesan dari artikel 1 dengan kata *instances* (dalam *nanosecond*)

Untuk pengetesan selanjutnya akan digunakan artikel yang sama dengan perbedaan yang dicari adalah *substring* dari instances yaitu *stance* yang masih merupakan kalimat ke 602.

No.	KMP	BM
1	420724	218813
2	423377	271193
3	414298	223702
4	447962	232013
5	412902	214762
6	415416	219441
7	415346	226984
8	426032	216648
9	456622	216577
10	415485	245003

Tabel 3.2 Pengetesan 2: Hasil Pengetesan dari artikel 1

dengan kata *stance* (dalam *nanosecond*)

Berikutnya akan diadakan pengetsan pada artikel yang sama dengan kata *greater* yang merupakan kata ke 89 dari artikel tersebut.

No.	KMP	BM
1	94006	91772
2	88349	89397
3	93797	97917
4	87231	90026
5	89677	93378
6	90445	93588
7	87650	97988
8	93797	106717
9	92260	90794
10	102597	91212

Tabel 3.3 Pengetesan 3: Hasil Pengetesan dari artikel 1 dengan kata *greater* (dalam *nanosecond*)

Pengetesan berikutnya akan digunakan pencarian kalimat yaitu kalimat *Eckhart's claims about CIQ coincide with widespread* yang dimulai dari kata ke 427 dalam artikel 1 tersebut.

No.	KMP	BM
1	274966	100362
2	264977	93727
3	267353	129974
4	267073	144013
5	271403	92260
6	296756	95473
7	272940	107277
8	274825	109161
9	264628	111956
10	281460	97010

Tabel 3.4 Pengetesan 4: Hasil Pengetesan dari artikel 1 dengan kalimat *Eckhart's claims about CIQ coincide with widespread* (dalam *nanosecond*)

Untuk pengetesan selanjutnya akan digunakan artikel 2 yang diambil dari:
http://www.appleinsider.com/articles/11/12/08/google_exec_expects_majority_of_tvs_to_have_google_tv_in_2012.html

Artikel tersebut memiliki panjang 563 kata dengan 3492 karakter. Pengetesan tetap dilakukan sebanyak sepuluh kali dengan kata yang dicari adalah kata *stepping* yang merupkan kata ke 557.

No.	KMP	BM
1	348019	179772
2	389644	176000
3	360591	156235
4	348857	148761
5	343828	157562
6	371625	163289
7	346622	156514
8	358705	218953
9	363664	168876
10	345155	157423

Tabel 3.5 Pengetesan 5: Hasil Pengetesan dari artikel 2 dengan kata *stepping* (dalam *nanosecond*)

Untuk pengetesan berikutnya akan digunakan kata *step* yang merupakan *substring* dari *stepping*.

No.	KMP	BM
1	359822	240045
2	349835	262742
3	362965	235785
4	357797	256318
5	347321	238229
6	345016	241441
7	358496	238438
8	348997	239556
9	346693	241721
10	348997	238788

Tabel 3.6 Pengetesan 6: Hasil Pengetesan dari artikel 2 dengan kata *step* (dalam *nanosecond*)

Sekali lagi Pengetesan akan dilakukan dengan artikel yang sama dengan pencarian kata *renew* yang ditemukan di kata ke 14.

No.	KMP	BM
1	28006	51054
2	27936	54896
3	33733	55384
4	34502	61530
5	26889	53638
6	30660	52800
7	35060	52032
8	26749	56361
9	26819	72775
10	33663	55454

Tabel 3.7 Pengetesan 7: Hasil Pengetesan dari artikel 2 dengan kata *renew* (dalam *nanosecond*)

Pengetesan selanjutnya akan dicari kalimat *Apple's rivals are preparing for the competition* dalam artikel ke 2 yang dimulai dari kalimat ke 436.

No.	KMP	BM
1	269867	93099
2	276362	113562
3	281530	88000

4	276222	106019
5	279435	90724
6	267213	93098
7	266374	92331
8	266235	94006
9	276152	90375
10	337403	88559

Tabel 3.8 Pengetesan 8: Hasil Pengetesan dari artikel 2 dengan kalimat *Apple's rivals are preparing for the competition* (dalam *nanosecond*)

3.1 Penggunaan dalam Bahasa Indonesia

Untuk penggunaan di Bahasa Indonesia masih menggunakan metode yang sama untuk pengetesan berikut akan digunakan artikel 3 yang diambil dari: http://tekno.kompas.com/read/2011/12/08/1804261/Tahun.Depan.Pendapatan.Operator.Diprediksi.Turun

Artikel tersebut memiliki panjang 300 kata dengan 4148 karakter. Kata yang dicari adalah *mengakuisisi* yang merupakan kata ke 289.

No.	KMP	BM
1	212317	110000
2	219092	100502
3	215879	93518
4	219441	118451
5	212527	106578
6	215390	95054
7	213365	95263
8	220559	94635
9	215390	209524
10	214412	93936

Tabel 3.9 Pengetesan 9: Hasil Pengetesan dari artikel 3 dengan kata *mengakuisisi* (dalam *nanosecond*)

Selanjutnya pengetesan akan dilakukan dengan kata *sisi* yang merupakan substring dari kata *mengakuisisi*.

No.	KMP	BM
1	211898	161753
2	219791	160425
3	228521	153162
4	215390	158120
5	218115	171670
6	218533	164267
7	214273	179213
8	218394	152533
9	214134	158888
10	217346	161752

Tabel 3.10 Pengetesan 10: Hasil Pengetesan dari artikel 3 dengan kata *sisi* (dalam *nanosecond*)

Pengetesan berikutnya akan menggunakan kata *beragam* yang merupakan kata ke 24 dari artikel tersebut.

No.	KMP	BM
1	33594	59993
2	39600	53429
3	39251	53918
4	36806	59016
5	32826	54686
6	33454	52940
7	34362	54127
8	37644	53987
9	37924	53708
10	36807	53569

Tabel 3.11 Pengetesan 11: Hasil Pengetesan dari artikel 3 dengan kata *beragam* (dalam *nanosecond*)

Selanjutnya akan dicari kalimat *Kami akan terus melakukan sosialisasi agar pengguna* yang ditemukan di kata ke 140 dari artikel.

No.	KMP	BM
1	115727	66977
2	116007	64953
3	114260	66279
4	112724	68025
5	113492	69283
6	114051	68235
7	112165	68584
8	111676	68933
9	112864	65930
10	112235	66000

Tabel 3.12 Pengetesan 12: Hasil Pengetesan dari artikel 3 dengan kalimat *Kami akan terus melakukan sosialisasi agar pengguna* (dalam *nanosecond*)

Berikutnya akan digunakan artikel 4 yang diambil dari: http://tekno.kompas.com/read/2011/12/08/13241922/Kalah.di.Pengadilan.Apple.Dilarang.Pakai.Merek.iPad.

Artikel tersebut berisi 254 kata dengan 1874 karakter. Pengetesan selanjutnya akan dicari kata *bernegosiasi* yang merupakan kata ke 210.

No.	KMP	BM
1	165524	87093
2	166292	83600
3	162660	82762
4	160426	84787
5	166013	85346
6	163568	85346
7	159308	81994
8	161054	83530
9	166152	81575
10	171181	85904

Tabel 3.13 Pengetesan 13: Hasil Pengetesan dari artikel 4 dengan kata bernegosiasi(dalam *nanosecond*)

Pengetesan berikutnya akan dilakukan dengan kata *nego* yang merupakan *substring* dari kata *bernegosiasi.*

No.	KMP	BM
1	165524	135632
2	166292	132000
3	162660	136330
4	160426	128997
5	166013	132769
6	163568	129974
7	159308	132978
8	161054	133816
9	166152	130813
10	171181	130953

Tabel 3.14 Pengetesan 14: Hasil Pengetesan dari artikel 4 dengan kata bernegosiasi(dalam *nanosecond*)

Setelah ini akan dilakukan pengetesan dengan kata *menolak* yang merupakan kata ke 84 dari artikel tersebut.

No.	KMP	BM
1	74381	72285
2	73962	70959
3	73263	75010
4	72635	70959
5	75219	72285
6	78013	69353
7	75708	74800
8	79899	73543
9	78432	69981
10	74032	71308

Tabel 3.15 Pengetesan 15: Hasil Pengetesan dari artikel 4 dengan kata *menolak*(dalam *nanosecond*)

Sama seperti berikutnya pengetesan selanjutnya akan dilakukan dengan kalimat yaitu *Apple sudah memiliki citra yang baik* yang ditemukan di kata ke 135.

No.	KMP	BM
1	127740	69562
2	117334	69073
3	116076	68654
4	117612	68794
5	114889	67816
6	114540	68235
7	115028	69422
8	120546	67327
9	119428	69492
10	117124	66978

Tabel 3.16 Pengetesan 16: Hasil Pengetesan dari artikel 4 dengan kalimat *Apple sudah memiliki citra yang baik* (dalam *nanosecond*)

IV. ANALISIS

Melihat hasi pengetesan dari data diatas dapat diambil hasil perhitungan sebgai berikut

Pengetesan 1:

	KMP	BM
Rata-rata	443471.2	178640.2
Rata-rata per kata	723.4440457	291.4195759
Rata-rata per karakter	101.4112051	40.85072033
Perbandingan	2.4	1

Tabel 4.1:Perhitungan pengetesan 1

Pengetesan 2:

	KMP	BM
Rata-rata	424816.4	228513.6
Rata-rata per kata	693.0120718	372.7791191
Rata-rata per karakter	97.14530071	52.25556826
Perbandingan	1.8	1

Tabel 4.2:Perhitungan pengetesan 2

Pengetesan 3:

	KMP	BM
Rata-rata	91980.9	94278.9
Rata-rata per kata	150.0504078	153.7991843
Rata-rata per karakter	21.03382118	21.55931855
Perbandingan	1	1.02

Tabel 4.3:Perhitungan pengetesan 3

Pengetesan 4:

	KMP	BM
Rata-rata	273638.1	108121.3
Rata-rata per kata	446.3916803	176.3805873
Rata-rata per karakter	62.57445689	24.72474274
Perbandingan	2.5	1

Tabel 4.4:Perhitungan pengetesan 4

Pengetesan 5:

	KMP	BM
Rata-rata	357671	168338.5
Rata-rata per kata	635.294849	299.0026643
Rata-rata per karakter	102.4258305	48.20690149

Perbandingan	2.1	1

Tabel 4.5:Perhitungan pengetesan 5

Pengetesan 6:

	KMP	BM
Rata-rata	352593.9	243306.3
Rata-rata per kata	626.2769094	432.1603908
Rata-rata per karakter	100.9719072	69.67534364
Perbandingan	1.4	1

Tabel 4.6:Perhitungan pengetesan 6

Pengetesan 7:

	KMP	BM
Rata-rata	30401.7	56592.4
Rata-rata per kata	53.99946714	100.5193606
Rata-rata per karakter	8.706099656	16.20630011
Perbandingan	1	1.8

Tabel 4.7:Perhitungan pengetesan 7

Pengetesan 8:

	KMP	BM
Rata-rata	279679.3	94977.3
Rata-rata per kata	496.7660746	168.698579
Rata-rata per karakter	80.09143757	27.19853952
Perbandingan	2.9	1

Tabel 4.8:Perhitungan pengetesan 8

Dari data pengetesan berBahasa Inggris di atas dapat dilihat bahwa secara umum algoritma *Boyer-Moore* lebih efektif dibanding algoritma *KMP* akan tetapi melihat perbandingan antara pengetesan 1 dan 5 dengan pengetesan 2 dan 6 dapat dilihat kinerja algoritma *Boyer-Moore* berkurang ketika kata yang dicari pendek.

Selain itu, menurut pengetesan 3 dan 7, algoritma *KMP* jauh lebih efektif daripada algoritma *Boyer-Moore* apabila kata ditemukan di awal atau artikel yang perlu dicari relatif pendek. Menurut pengetesan 4 dan 8, algoritma *Boyer-Moore* juga jauh lebih efektif apabila yang dicari adalah kalimat.

Bedasarkan rata-rata perbandingan di atas dan juga perbandingan jumlah karakter per kalimat sebesar 6.7(dihitung dari artikel diatas) penggunaan algoritma *Boyer-Moore* pada Bahasa Inggris lebih efektif 41% daripada algoritma *Knuth-Morris-Pratt.*

Pengetesan 9:

	KMP	BM
Rata-rata	215837.2	111746.1
Rata-rata per kata	719.4573333	372.487
Rata-rata per karakter	52.0340405	26.9397541
Perbandingan	1.9	1

Tabel 4.9:Perhitungan pengetesan 9

Pengetesan 10:

	KMP	BM
Rata-rata	217639.5	162178.3
Rata-rata per kata	725.465	540.5943333
Rata-rata per karakter	52.46853905	39.09795082
Perbandingan	1.3	1

Tabel 4.10:Perhitungan pengetesan 10

Pengetesan 11:

	KMP	BM
Rata-rata	36226.8	54937.3
Rata-rata per kata	120.756	183.1243333
Rata-rata per karakter	8.733558341	13.2442864
Perbandingan	1	1.5

Tabel 4.11:Perhitungan pengetesan 11

Pengetesan 12:

	KMP	BM
Rata-rata	113520.1	67319.9
Rata-rata per kata	378.4003333	224.3996667
Rata-rata per karakter	27.36743009	16.22948409
Perbandingan	1.6	1

Tabel 4.12:Perhitungan pengetesan 12

Pengetesan 13:

	KMP	BM
Rata-rata	164217.8	84193.7
Rata-rata per kata	646.5267717	331.4712598
Rata-rata per karakter	87.62956243	44.92726788
Perbandingan	1.9	1

Tabel 4.12:Perhitungan pengetesan 13

Pengetesan 14:

	KMP	BM
Rata-rata	164217.8	132426.2
Rata-rata per kata	646.5267717	521.3629921
Rata-rata per karakter	87.62956243	70.66499466
Perbandingan	1.2	1

Tabel 4.14:Perhitungan pengetesan 14

Pengetesan 15:

	KMP	BM
Rata-rata	75554.4	72048.3
Rata-rata per kata	297.4582677	283.6547244
Rata-rata per karakter	40.3171825	38.44626467
Perbandingan	1.04	1

Tabel 4.15:Perhitungan pengetesan 15

Pengetesan 16:

	KMP	BM
Rata-rata	118031.7	68535.3
Rata-rata per kata	464.6917323	269.8240157
Rata-rata per karakter	62.98383138	36.57166489
Perbandingan	1.7	1

Tabel 4.16:Perhitungan pengetesan 16

Secara umum hasil pengetesan 9 sampai 16 menunjukan hasil yang sama dalam efektivitas kedua algoritma tersebut. Akan tetapi di dalam Bahasa Indonesia, algoritma *Boyer-Moore* 23% lebih efektif dibandingkan algoritma *Knuth-Morris-Pratt*, walaupun rata–rata karakter per kata di Bahasa Indonesia sebesar 10.61(dihitung dari artikel diatas) hal ini bertentangan dengan pengetesan diatas yang menunjukan *Boyer-Moore* yang lebih efektif bila kalimat lebih panjang. Hal ini mungkin disebabkan kosakata di Bahasa Inggris lebih mendukung *character jump* yang ada di *Boyer-Moore*. Sedangkan kosakata Bahasa Indonesia lebih mendukung *border function* yang ada di algoritma *Knuth-Morris-Pratt*.

Menurut perhitungan rata-rata diatas walaupun terdapat sedikit perbedaan secara umum algoritma *Boyer-Moore* lebih efektif 32% daripada algoritma *Knuth-Morris-Pratt.*
Sehingga *Boyer-Moore* akan lebih efektif apabila digunakan dalam kehidupan sehari-hari.

V. KESIMPULAN

1. Secara umum, algoritma *Boyer-Moore* lebih efektif daripada algoritma *Knuth-Morris-Pratt.*
2. Apabila teks yang dicari pendek algoritma *Knuth-Morris-Pratt* lebih efektif daripada algoritma *Boyer-Moore.*
3. Efektivitas algoritma *Boyer-Moore* tergantung panjang kata atau yang dicari.
4. Efektivitas algoritma pencarian string dapat tergantung dari bahasa teks atau artikel yang diperlukan pencarian.
5. Parameter seperti panjang karakter per kata dan pengulangan karakter dalam kata dapat mempengeruhi efektivitas algoritma pencarian *string.*

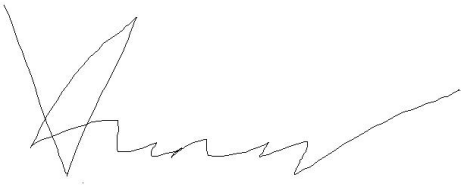
DAFTAR PUSTAKA

[1] http://www.informatika.org/~rinaldi/Stmik/2011-2012/PatternMatching2.ppt
[2] http://en.wikipedia.org/wiki/Knuth–Morris–Pratt_algorithm
[3] http://en.wikipedia.org/wiki/Boyer_moore
[4] http://www.upi.com/Business_News/Security-Industry/2011/12/07/Smartphone-probe-exposes-computer-software-and-hardware-risks/UPI-25881323292066/?spt=hs&or=si
[5] http://www.appleinsider.com/articles/11/12/08/google_exec_expects_majority_of_tvs_to_have_google_tv_in_2012.html
[6] http://tekno.kompas.com/read/2011/12/08/1804261/Tahun.Depan.Pendapatan.Operator.Diprediksi.Turun
[7] http://tekno.kompas.com/read/2011/12/08/13241922/Kalah.di.Pengadilan.Apple.Dilarang.Pakai.Merek.iPad.
[8] http://en.wikipedia.org/wiki/String_searching_algorithm

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, 9 Desember 2011



Kevin Wibowo - 13509065