BAB III
ANALISIS ALGORITMA

III.1 Analisis Sistem

Bab ini akan membahas tentang analisis dan perancangan sistem algoritma *knapsack* dengan data proyek yang digunakan bersumber dari PT. GITS Indonesia. Terdapat langkah-langkah yang dilakukan dalam analisis sistem yaitu :

1. Analisis masalah
2. Analisis struktur data
3. Analisis strategi
4. Analisis algoritma

III.1.1 Analisis Masalah

Analisis algoritma *knapsack* dilakukan karena ditemukannya masalah dalam penggunaan algoritma *knapsack*. Masalah yang ditemukan saat analisis dilakukan antara lain adalah belum ditemukannya analisis performansi algoritma *knapsack* terhadap kasus pemilihan proyek yang terjadi di PT. GITS Indonesia. Hal ini menyebabkan hanya pada kasus-kasus tertentu saja mungkin suatu algoritma lebih efektif dari yang lainnya. Penerapan Algoritma *knapsack* terhadap pemilihan proyek ini di PT. GITS Indonesia ini memiliki kesamaan karateristik dengan kasus uji yang dilakukan peneliti lain terhadap algoritma *knapsack* terutama pada parameter yang digunakan. Kapasitas *knapsack* (*W*) diasumsikan dengan jumlah sumber daya manusia yang tersedia di perusahaan, bobot (*wi*) diasumsikan dengan jumlah sumber daya manusia yang dibutuhkan untuk menangani sebuah proyek dan profit (*pi*) diasumsikan dengan keuntungan dari sebuah proyek.

Hasil penelitian yang dilakukan terhadap beberapa literatur ditemukan bahwa algoritma *knapsack* rata-rata digunakan untuk menemukan solusi optimal yaitu maksimasi atau minimasi. Solusi yang dihasilkan dari algoritma *knapsack* untuk kasus pemilihan proyek di PT. GITS Indonesia adalah optimalitas pemilihan proyek. Pada kasus pemilihan proyek di PT. GITS Indonesia dapat ditentukan *constraintnya,* yaitu berupa jumlah sumber daya manusia yang tersedia di PT. GITS Indonesia, sedangkan parameter uji lainnya adalah jumlah sumber

daya manusia yang dibutuhkan per proyek dan profit proyek yang ditangani. Berdasarkan analisis masalah yang dipaparkan diatas maka diperlukan analisis performansi terhadap algoritma *knapsack*.

III.1.2 Analisis Struktur Data

Struktur data yang digunakan didalam penelitian ini adalah struktur data *array* dinamis. Parameter yang menggunakan *array* dinamis antara lain adalah :

1. Parameter profit (profit : Larik Of Int)
2. Parameter sumber daya manusia (SDM : Larik Of Int)
3. Parameter density (density : Larik Of Double)
4. Parameter status proyek (optimum : Larik Of Double)

Parameter diatas menggunakan *array* dinamis dengan jumlah elemen sebesar max, dimana max ini adalah parameter yang dapat dirubah pada saat *runtime* program. Banyaknya elemen dipanggil dan ditentukan dengan menggunakan setlength. Pemanggilan setLength untuk setiap parameter dalam potongan program adalah sebagai berikut :

1. setLength(SDM,max)
2. setLength(profit,max)
3. setLength(density,max)
4. setlength(optimum,max)

III.1.3 Analisis Strategi

Persoalan pemilihan proyek dalam penelitian ini akan diselesaikan dengan strategi algoritma *Greedy*. Sebelum kita menganalisis algoritma *knapsack* maka kita harus menganalisis algoritma *greedy* terlebih dahulu. Adapun karateristik algoritma *Greedy* yaitu :

1. Tidak melihat masalah secara global.
2. Mengambil langkah terbaik pada setiap langkah pencariannya untuk segera mendapatkan solusi.
3. Berguna dalam banyak kasus dimana tujuan atau kendala tidak diketahui secara pasti.
4. Umumnya menggunakan perhitungan kompleksitas waktu O(n).

5. Mudah untuk menerapkan dan penginterprestasikan hasil.

Seperti yang telah dijelaskan pada bab tinjauan pustaka strategi algoritma greedy memiliki tiga pendekatan yang berbeda, yaitu *greedy by weight*, *greedy by profit* dan *greedy by density*. Percobaan perhitungan pendekatan strategi algoritma greedy dilakukan untuk menentukan pemilihan pendekatan yang akan digunakan dalam kasus pemilihan proyek. perbandingan perbandingan pendekatan algoritma greedy akan disajikan dalam Tabel III-1. Diketahui data proyek sebagai berikut :

1) Jumlah sumber daya manusia yang tersedia di perusahaan (W) = 10.
2) Jumlah projek yang akan dikerjakan (i) = 6
3) (wi) = weight atau jumlah SDM yang dibutuhkan untuk pengerjaan proyek.
4) (pi) = profit.

Tabel III-1 Perhitungan Perbandingan Pendekatan Algoritma *Greedy* dengan Fractional Knapsack Ke-1

Properti Proyek				***Fractional Knapsack With Greedy by***		
i	w_i	p_i	p_i/w_i	*profit*	*Weight*	*density*
1	2	50000	25000	1/2 or 50%	1	1
2	3	51000	17000	1	2/3 or 66%	0
3	2	45000	22500	0	1	1
4	4	123000	30750	1	0	1
5	2	40000	20000	0	1	0
6	2	56000	28000	1	1	1
Total SDM Terpakai				**10**	**10**	**10**
Total keuntungan				**255000**	**225000**	**274000**

Pada Tabel III-1 dapat dibandingkan bahwa pendekatan strategi algoritma *greedy by density* menghasilkan keuntungan paling baik. Total keuntungan untuk greedy by density adalah 274000. Angka 0 (nol) pada kolom tabel *fractional knapsack* diatas menunjukkan proyek ke-i tidak terpilih sebagai solusi. Angka 1 (satu) menunjukkan proyek ke-i terpilih sebagai solusi sedangkan

angka pecahan (/) atau persen (%) menunjukkan proyek ke-i terpilih sebagai solusi optimal dengan pertimbangan pecahan atau persentase tersebut.

Perhitungan perbandingan algoritma greedy akan dilakukan sekali lagi untuk memastikan kebaikan strategi algoritma yang akan digunakan untuk pemilihan proyek. perhitungan perbandingan algoritma greedy ke-2 dapat dilihat pada tabel Tabel III-2.

Tabel III-2 Perhitungan Perbandingan Pendekatan Algoritma *Greedy* dengan *Fractional Knapsack* Ke-2

Properti Proyek				***Fractional Knapsack With Greedy by***		
i	w_i	p_i	p_i/w_i	***profit***	***weight***	***density***
1	2	10000	5000	0	1	1
2	3	45000	15000	1	1	1
3	2	37250	18625	1	1	1
4	3	12500	4166,66667	67%	1	0
5	3	41600	13866,6667	1	0	1
Total SDM Terpakai				**10**	**10**	**10**
Total keuntungan				**50269,44**	**42791,67**	**52491,67**

Hasil perhitungan pada Tabel III-2 menunjukkan bahwa pendekatan strategi algoritma *greedy by density* lagi-lagi menghasilkan keuntungan paling optimal. Total keuntungan untuk pendekatan strategy algoritma *greedy by density* adalah 52491,67. Berdasarkan hasil percobaan tersebut maka pendekatan strategi algoritma yang akan digunakan dalam pemilihan proyek di PT GITS Indonesia adalah dengan menggunakan *greedy by density*. Teori ini diperkuat dengan teori yang dikemukkan Rinaldi Munir yaitu algoritma *greedy* untuk persoalan *fractional knapsack* dengan strategi pemilihan objek berdasarkan density (p_i/w_i) terbesar akan selalu memberikan solusi optimal [**8**].

Penjelasan langkah atau proses algoritma *greedy* dalam mencari solusi akan dijelaskan untuk mempermudah pemahaman langkah algoritma. Proses diilustrasikan dengan penukaran uang W=500 yang akan ditukar dengan himpunan nilai uang 100, 50, 100, 200, 100, 50.

1. Menginisialisasikan constraint dan himpunan kandidat.

2. Langkah pertama yang dilakukan algoritma *greedy* adalah menyusun himpunan-himpunan yang mempresentasikan solusi optimal. Contoh himpunan koin yang mempresentasikan solusi optimal adalah K={50, 50, 100, 100, 100, 200}.
3. Langkah kedua yang dilakukan adalah memilih solusi yang nilainya mendekati atau sama dengan solusi optimal. Contoh himpunan yang mendekati solusi optimal adalah S={200, 100, 100, 100} atau S={200, 100, 100, 50, 50}.
4. Langkah berikutnya yaitu menyeleksi himpunan yang nilainya paling tinggi dari himpunan kandidat yang tersisa.
5. Langkah keempat yaitu memeriksa total nilai himpunan solusi yang diseleksi. Total nilai himpunan solusi harus lebih kecil atau sama dengan *constrain* yang telah ditentukan.

$$C = 500$$
$$S = 200 + 100 + 100 + 100$$
$$= 500$$
$$S \leq C$$
$$500 \leq 500$$

   Jika hasil nilai diatas *true* maka himpunan proyek terpilih akan diseleksi menjadi solusi optimal.
6. Langkah terakhir adalah menyusun nilai solusi yang paling optimal.

Algoritma *greedy* mencari solusi terbaik dengan cara memilih himpunan yang dianggap paling baik, sehingga pilihan ini disebut sebagai solusi lokal. Solusi lokal yang dipilih diharapkan dapat mengarah ke solusi global.

III.1.4 Analisis Algoritma *knapsack*

Algoritma *fractional knapsack* merupakan algoritma yang digunakan untuk menyelesaikan permasalahan pemilihan proyek. Pada Gambar III-1 Flowchart Algoritma *Knapsack* dapat dilihat alur proses algoritma *Knapsack*.

Adapun langkah-langkah algoritma *fractional knapsack* dalam mencari solusi optimal yaitu :

1. Memasukkan harga ke dalam SDM[i], profit[i] dan k.

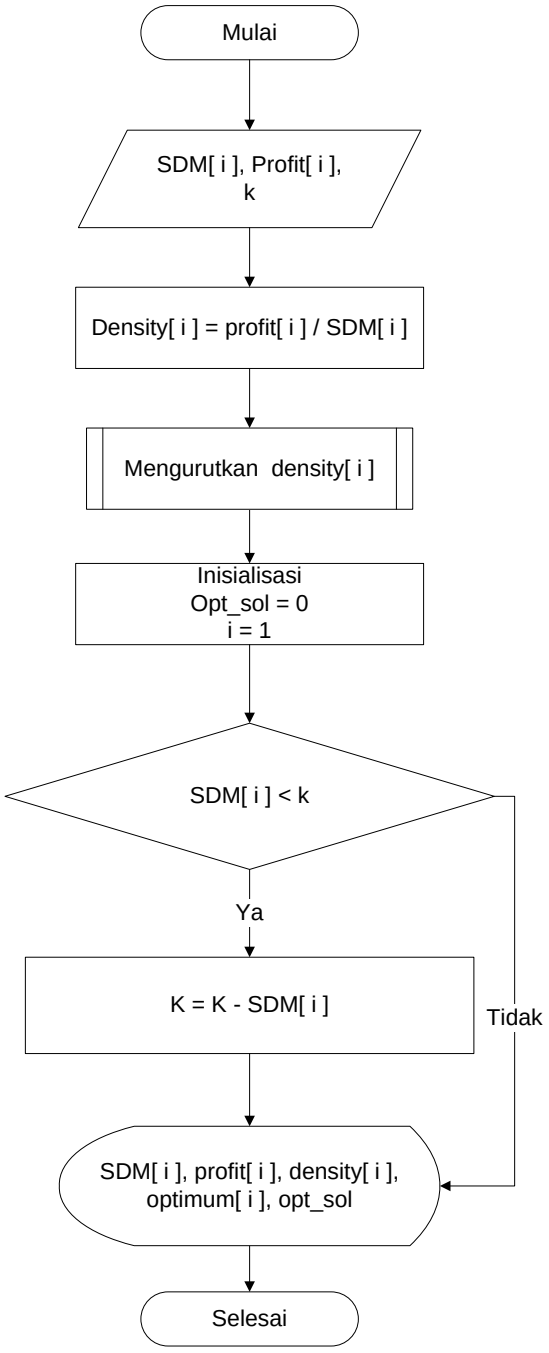
   SDM [i] adalah jumlah SDM yang dibutuhkan untuk sebuah proyek, p[i] adalah profit proyek dan k adalah *constraint* atau jumlah SDM yang tersedia di perusahaan.
2. Hitung rasio *density* atau Pi/Wi (keuntungan per unit) untuk setiap proyek.
3. Urutkan proyek secara menurun berdasarkan rasio density proyek.
4. Inisialisasi opt_sol = 0 dan i=1.
5. Pilih proyek dengan keuntungan terbesar tanpa melampaui jumlah SDM yang tersedia di perusahaan.

```
for i ← to n do
  if SDM[i] > k then
    eof
  else
    optimum[i] ← 1
     k←k-SDM[i]
  endif
  if i<= n then optimum[i] ← k/SDM[i]
    else opt_sol←0
```

6. Tampilkan daftar proyek dengan solusi yang terbaik.

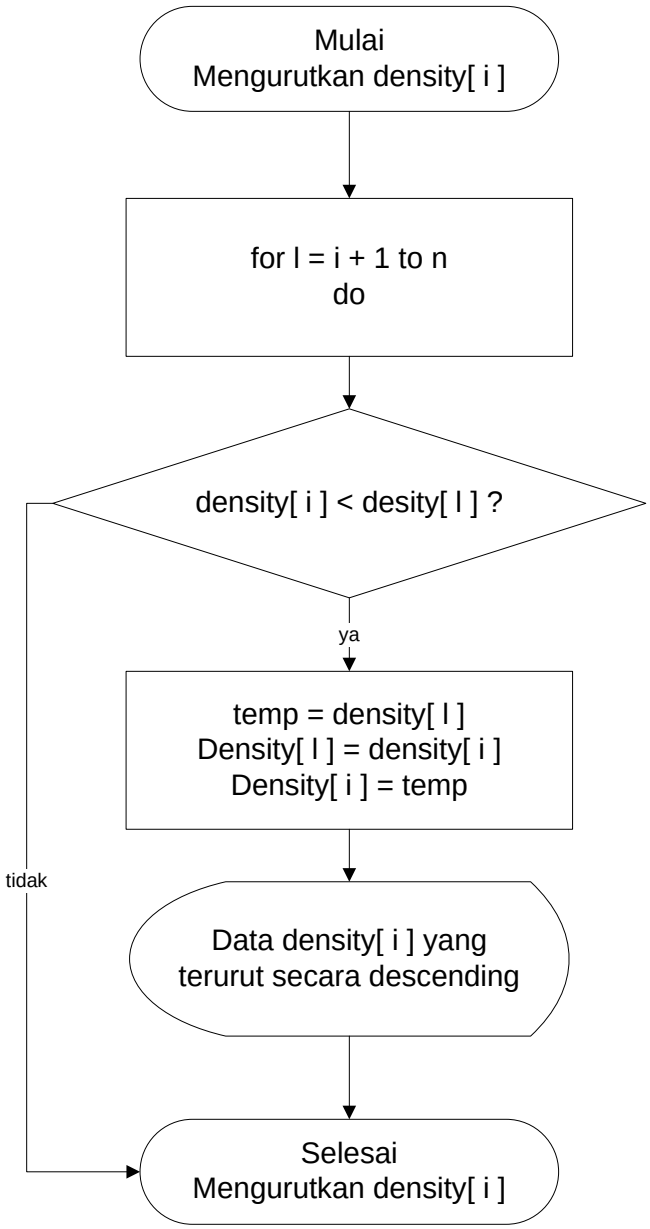
Ketika proses pemilihan solusi, algoritma *knapsack* menginisialisasikan jumlah SDM yang dibutuhkan untuk proyek, keuntungan proyek dan jumlah SDM yang tersedia di perusahaan. Setelah inisialisasi himpunan penyusun solusi algoritma knapsack akan menghitung rasio density (*pi* / *wi*) proyek. Proses berikutnya algoritma akan malakukan pengurutan secara *descending* (menurun) berdasarkan *density*.

Tahapan berikutnya adalah inisialisasi nilai solusi optimal dan inisialisasi iterasi. Setelah itu baru algoritma kapsack mulai menyeleksi proyek dengan nilai *density* terbaik, penyeleksian akan dilakukan tanpa melebihi jumlah SDM yang tersedia di perusahaan. Jika jumlah total SDM telah memenuhi atau kurang dari sama dengan *constraint* maka himpunan proyek yang telah diseleksi akan disusun dan ditampilkan dalam output algoritma *knapsack*. Proses algoritma knapsack tersebut dapat dilihat pada Gambar III-1 Flowchart Algoritma *Knapsac*k.



Gambar III-1 Flowchart Algoritma *Knapsack*

Prosedur mengurutkan nilai berdasarkan *density* akan dipanggil dalam pseudo-code algoritma *knapsack* untuk mengurutkan nilai *density* proyek secara menurun (*denscending*). Alur proses pengurutan dapat dilihat pada Gambar III-2 Flowchart Prosedur Mengurutkan Nilai Berdasarkan *Density*.



Gambar III-2 Flowchart Prosedur Mengurutkan Nilai Berdasarkan *Density*

Prosedur pengurutan nilai density dapat dilihat dalam Tabel III-3. Pengurutan dilakukan untuk parameter array density. Prosedur pengurutan ini akan dipanggil dalam badan algoritma utama *pseudo-code* algoritma *knapsck*. *Pseudo-code* algoritma fractional knapsack dapat dilihat pada Tabel III-4. Dari *Pseudo-code* tersebut ditandai algoritma inti *knapsacknya* untuk kemudian dilakukan perhitungan kompleksitas waktunya.

Tabel III-3 *Pseudo-Code* Pengurutan Algoritma *Knapsack*

```
1   procedure PengurutanNilaiBerdasarkanDensity(input
2   density[i]:Larik of double)
3   {mengurutkan nilai proyek berdasarkan density proyek secara
4   menurun}
5   {I.S : nilai density telah terdefinisi sebelumya}
6   {F.S : menampilkan pengurutan density proyek}
7
8   Algoritma :
9   //Pengurutan berdasarkan density
10  i←1
11    for l←i+1 to n do
12      if (density[i]<density[l]) then
13         temp←density[l]
14         density[l]←density[i]
15         density[i]←temp
16      endif
17  endfor
18  write(density[i])
19
```

Tabel III-4 *Pseudo-code* Utama Algoritma *Fractional Knapsack*

```
1   procedure Fractional Knapsack(input SDM:Larik of Int,
2   profit:Larik of Int, output density:Larik of Double,
3   optimum:Larik of Double)
4
5   {IS:Penyelesaian persoalan fractional knapsack dengan
6   algoritma greedy menggunakan strategi pemilihan proyek
7   berdasarkan density (pi/wi)
8   FS:Menghasilkan output berupa list proyek yang memiliki
9   keuntungan terbesar dan total bobot proyek tidak boleh
10  melebihi kapasitas knapsack}
11
12  Kamus
13      opt_sol           : real {total profit optimal}
14      k,n,i,temp, temp2 : integer {k=constraint/jumlah sdm
15  yang tersedia di perusahaan, n=data ke-n, i=iterasi}
16
17  procedure PengurutanNilaiBerdasarkanDensity(input Larik Of
18  Double : density[i])
19
20  Algoritma:
21
22  read (SDM[i], profit[i], k)
23  density[i]←profit[i]/SDM[i]
24  PengurutanNilaiBerdasarkanDensity(density[i])

25  Opt_sol←0
26  i←1
27  for i←1 to n do
28     optimum[i] ← 0
29  endfor
30  ------------------------------------------------------------
31   for i ← to n do                                    O(n)
32     if SDM[i] > k then                               O(1)
```

33	eof	O(1)
34	else	O(1)
35	optimum[i] ← 1	O(1)
36	k←k-SDM[i]	O(1)
37	endif	O(1)
38	if i<= n then optimum[i] ← k/SDM[i]	O(1)
39	else opt_sol←0	O(1)
40		
41	for i←0 to n do	
42	opt_sol←opt_sol+(profit[i]*optimum[i])	
43	endfor	
44	endif	
45	endfor	
46		
47	write (SDM[i], profit[i], density[i], optimum[i], opt_sol)	

Perhitungan kompleksitas waktu dengan perhitungan Big-O dilakukan dengan menghitung algoritma inti pemilihan *knapsack* dan mengabaikan proses pengurutan yang dilakukan algoritma. Adapun algoritma inti ditandai dengan kotak yang terdapat di dalam tabel *pseudo-code*. Perhitungan kompleksitas waktu ditunjukkan dalam Tabel III-5 Perhitungan Big-O Algoritma *Fractional Knapsack*.

Tabel III-5 Perhitungan Big-O Algoritma *Fractional Knapsack*

	Pseudo-code	**Nilai Big-O**
1	for i ← to n do	O(n)
2	if SDM[i] > k then	O(1)
3	eof	O(1)
4	Else	O(1)
5	optimum[i] ← 1	O(1)
6	k←k-SDM[i]	O(1)
7	Endif	O(1)
8	if i<= n then optimum[i] ← k/SDM[i]	O(1)
9	Else opt_sol ← 0	O(1)
10	**Jumlah**	**O(n)**

Berdasarkan hasil perhitungan performansi algoritma *fractional knapsack* dengan menggunakan notasi Big-O didapat kompleksitas waktunya yaitu **O(n)**, dengan n adalah ukuran inputan.

Contoh data proyek dapat dilihat dalam Tabel III-6 Sample Data Proyek 1. Data dikategorikan berdasarkan nama projek. Data sample yang didapat memiliki

atribut jumlah sumber daya manusia yang dibutuhkan untuk projek tersebut dan budget dari tiap projek.

Tabel III-6 Sample Data Proyek 1

No	Nama Proyek	Kebutuhan SDM	Profit (By Rasio)
1	Proyek A	2	15300
2	Proyek B	2	23000
3	Proyek C	2	10500
4	Proyek D	4	26950
5	Proyek E	2	24650
6	Proyek F	2	12425

Kolom nama proyek pada sample data menunjukkan nama-nama proyek syang akan dikerjakan, sedangkan pada kebutuhan SDM terdapat bobot atau dalam kasus ini adalah banyak SDM yang diperlukan untuk mengerjakan proyek. Sedangkan untuk kolom profit terdapat nilai keuntungan per proyek dalam satuan rasio proyek.

Setelah data diatas didapat kemudian dilakukan penentuan pemilhan projek menggunakan algoritma *Fractional Knapsack* dengan pendekatan *greedy by density*. Diketahui properti *knapsack* dengan ketentuan sebagai berikut :

1) Kapasitas *knapsack* (W) = 9.
2) Jumlah projek yang akan dikerjakan (i) = 6
3) (wi) = weight atau jumlah SDM yang dibutuhkan untuk pengerjaan proyek.
4) (pi) = profit.

W diilustrasikan sebagai sumber daya manusia yang tersedia di perusahaan.

Berdasarkan Tabel III-6 Sample Data Proyek 1 dan ketentuan diatas maka *fractional knapsack* akan menghasilkan perhitungan pemilihan proyek yang dapat dilihat dalam Tabel III-7 Sample Data Proyek 1, *Pemilihan Proyek* dengan Menggunakan *Fractional Knapsack dan Pendekatan Greedy By Density*.

Tabel III-7 Sample Data Proyek 1, Pemilihan Proyek dengan Menggunakan *Fractional Knapsack* dan Pendekatan *Greedy By Density*

Properti proyek				***Greedy by density***
i	w_i	p_i	p_i/w_i	
1	2	15300	7650	1
2	2	23000	11500	1
3	2	10500	5250	0
4	4	26950	6737,5	0,75 or 75%
5	2	24650	12325	1
6	2	12425	6212,5	0
Total bobot (Total SDM yang terpakai)				**9**
Total keuntungan				83.163
Kapasitas *knapsack* (jumlah SDM yang tersedia)				**9**

Pada Tabel III-7 dilakukan percobaan perhitungan *fractional knapsack* dengan pendekatan *greedy by density* dan telah berhasil memberikan solusi optimal. Nilai 1 (satu) dalam kolom *greedy by density* berarti proyek pada baris ke-i terpilih sebagai solusi optimal untuk kemudian dimasukkan ke dalam *knapsack*. Nilai 0 (nol) dalam kolom greedy by weight berarti proyek tidak terpilih sebagai solusi, sedangkan nilai 0,75 atau 75% berarti proyek dengan baris ke i=4 dipilih dengan pertimbangan 75% dapat memberikan keuntungan optimal.

Solusi optimal permasalahan ini adalah S = (1, 1, 0, 75%, 1, 0) dengan total keuntungan = 83.163. Agar hasil uji dengan sample data yang didapatkan lebih baik dan akurat maka diperlukan percobaan kedua. Salah satu cara agar hasil uji yang didapatkan lebih baik maka pengolahan sample yang dilakukan pada data kedua dilakukan dengan jumlah proyek yang berbeda. Sample data uji kedua disajikan dalam tabel Tabel III-8.

Tabel III-8 Sample Data Proyek 2

No	**Proyek (Project yang dikerjakan)**	**Weight (SDM)**	**Profit (By Ratio)**
1	Proyek G	2	10000
2	Proyek H	3	45000
3	Proyek I	2	37250

4	Proyek J	3	12500
5	Proyek K	3	41600

Data diatas adalah data sample proyek ke-2 kemudian dilakukan penentuan pemilhan projek menggunakan algoritma *Fractional Knapsack* dengan pendekatan *greedy by density*. Diketahui sumber daya manusia yang tersedia (kapasitas *knapsack*) adalah W = 9, kemudian jumlah proyek (project) yang akan dikerjakan sebanyak i = 5 dengan properti proyeknya adalah weight atau jumlah SDM yang dibutuhkan untuk pengerjaan proyek (wi) dan profit (pi). Maka dari Tabel III-8 Sample Data Proyek 2 akan menghasilkan perhitungan pemilihan proyek yang dapat dilihat dalam Tabel III-9.

Tabel III-9 Sample Data Proyek 2 Pemilihan Proyek dengan Menggunakan Fractional *Knapsack* dan Pendekatan *Greedy* By Density

Properti proyek				***Greedy by***
i	w_i	p_i	p_i/w_i	***density***
1	2	10000	5000	0,5 *or* 50%
2	3	45000	15000	1
3	2	37250	18625	1
4	3	12500	4166,66667	0
5	3	41600	13866,6667	1
Total bobot				**9**
Total keuntungan				**128.850**
Kapasitas *knapsack* (jumlah SDM yang tersedia)				**9**

Percobaan perhitungan *fractional knapsack* dapat dilihat pada Tabel III-9. Hasil percobaan dilakukan dengan pendekatan *greedy by density* dan telah berhasil memberikan solusi optimal. Solusi optimal permasalah ini adalah *S* = (50%, 1, 1, 0, 1) dengan total keuntungan = 128.850.

III.1.5 Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak dilakukan berdasarkan kebutuhan perangkat lunak pemilihan proyek untuk penerapan analisis agoritma serta hasil observasi. Spesifikasi kebutuhan perangkat lunak akan dibagi kedalam dua bagian

yaitu SKPL-F (Spesifikasi Kebutuhan Perangkat Lunak Fungsional) dan SKPL-NF (Spesifikasi kebutuhan perangkata lunak non-fungsional) Berikut ini adalah tabel spesifikasi kebutuhan perangkat lunak fungsional dan non-fungsional pada tabel Tabel III-10 Spesifikasi Kebutuhan Perangkat Lunak Fungsional dan Tabel III-11 Spesifikasi Kebutuhan Perangkat Lunak Non-Fungsional :

Tabel III-10 Spesifikasi Kebutuhan Perangkat Lunak Fungsional

Kode	**Kebutuhan**
SKPL-F001	Perangkat lunak dapat memasukan data proyek yang diinginkan oleh user.
SKPL-F002	Perangkat lunak dapat memproses hasil inputan untuk kemudian dimulai pemilihan solusi optimal.
SKPL-F003	Perangkat lunak dapat mengukur waktu eksekusi ketika memulai pemilihan solusi.
SKPL-F004	Perangkat lunak dapat menampilkan penggunaan ruang memori saat pemilihan solusi optimal.
SKPL-F005	Perangkat lunak dapat menampilkan hasil dari pemilihan solusi.

Tabel III-11 Spesifikasi Kebutuhan Perangkat Lunak Non-Fungsional

Kode	**Kebutuhan**
SKPL-NF001	Pengguna atau user yang menggunakan perangkat lunak ini adalah user yang ingin melakukan pemilihan proyek yang optimal.
SKPL-NF002	Perangkat lunak yang dibangun berbasis desktop.
SKPL-NF003	Perangkat keras yang digunakan adalah komputer dengan spesifikasi minimal processor Intel Pentium 4 2.0 GHz, memori 512 MB, keyboard, dan mouse.
SKPL-NF004	Bahasa pemograman yang digunakan adalah delphi

III.1.6 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional adalah langkah dimana seorang pembangun perangkat lunak (*software developer*) menganalisis sumber daya yang akan digunakan dan menggunakan perangkat lunak yang dibangun. Perangkat keras dan perangkat lunak yang dimiliki harus sesuai dengan kebutuhan atau permintaan, sehingga dapat ditentukan kompabilitas aplikasi yang dibangun terhadap sumber daya yang ada.

Setelah melakukan analisis non fungsional, maka dilanjutkan ke langkah berikutnya yaitu menentukan kebutuhan non fungsional sistem yang akan dibangun untuk disesuaikan dengan fakta yang ada.

Apabila terjadi ketidakcocokan antara fakta dan kebutuhan maka perlu dilakukan penyesuaian fakta terhadap kebutuhan yang ada. Apabila kebutuhan tidak dipenuhi maka sistem yang dibangun tidak akan berjalan baik sesuai yang diharapkan.

Analisis kebutuhan non fungsional yang dilakukan dibagi dalam tiga tahap, yaitu:

1. Analisis perangkat keras (hardware).
2. Analisis perangkat lunak (software).
3. Analisis pengguna (user).

III.1.6.1 Analisis Kebutuhan Perangkat Keras

Berdasarkan analisis perangkat keras yang sedang digunakan di PT GITS Indonesia, spesifikasi perangkat keras yang digunakan adalah sebagai berikut :

1. Processor 2.4 GHz
2. Monitor LCD 16 inch
3. VGA On-Board
4. Memori 1024 MB
5. Optical Drive DVD-ROM
6. Keyboard
7. Mouse

Spesifikasi perangkat keras minimum pada perangkat lunak yang akan dibangun pada unit personal komputer agar dapat menjalankan aplikasi secara optimal adalah sebagai berikut :

1. Processor 1.3 GHz
2. Monitor CRT 15 inch
3. VGA On-Board
4. Memori 128 MB
5. Optical Drive DVD-ROM
6. Keyboard
7. Mouse

Berdasarkan analisis spesifikasi perangkat keras yang ada di PT GITS Indonesia saat ini sudah memenuhi syarat untuk menerapkan sistem yang akan dibangun sehingga tidak memerlukan penambahan.

III.1.6.2 Analisis Kebutuhan Perangkat Lunak

Sistem operasi yang digunakan saat ini di PT GITS Indonesia menggunakan Windows 8, sedangkan kebutuhan perangkat lunak dalam membangun dan menerapkan sistem yang akan dibuat adalah sebagai berikut :

1. Sistem Operasi Windows XP
2. Borland Delphi 7

Spesifikasi kebutuhan perangkat lunak tersebut dipilih karena kemudahan implementasinya, familiar, interaktif dan mudah dalam memahami cara kerjanya.

III.1.6.3 Analisis Kebutuhan Perangkat Pikir

Analisis dan spesifikasi kebutuhan diperlukan agar kemampuan perangkat lunak yang dibangun menjadi jelas. Beberapa analisis dan kebutuhan yang berkaitan dengan perangkat lunak yang akan dibangun nanti yaitu analisis dan kebutuhan pengguna. Adapun karakteristik pengguna pada perangkat lunak pemilihan proyek yang akan dibangun yaitu untuk analis proyek PT GITS Indonesia yang akan melakukan pemilihan proyek. Adapun karakteristik dari pengguna adalah sebagai berikut :

1. Pengguna minimal mampu menggunakan keyboard sebagai sarana penginputan data proyek.

2. Pengguna dapat membaca

   Sedangkan untuk user yaitu sebagai berikut :

   1. Pengguna dapat menggunakan komputer, minimal mampu menggunakan keyboard sebagai sarana penginputan data.
   2. Jenjang pendidikan dimulai dari Anak sekolah (SD,SMP dan SMA), Mahasiswa.

Berdasarkan analisis pengguna, dapat diambil kesimpulan bahwa pengguna yang ada telah memenuhi syarat sebagai pengguna sistem yang akan dikembangkan, sehingga tidak diperlukan pelatihan khusus mengenai penggunaan sistem, cukup berupa dokumen atau buku panduan untuk membantu menjalankan perangkat lunak.