Penerapan *Brute Force*, Teori NP dan Pendekatan Algoritma *Greedy* untuk Penentuan Jalur Kritis dalam Diagram Jaringan Proyek

Jais Anasrulloh Ja'fari and 13511036
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13511036@std.stei.itb.ac.id

***Abstrak*—Algoritma *greedy* merupakan algoritma yang paling populer dan cukup sederhana dalam persoalan optimasi. Persoalan optimasi sendiri terdiri dari persoalan dalam mencari solusi paling maksimal atau persoalan dalam mencari solusi paling minimal.**

Algoritma NP(*Non-deterministic Polynomial*) adalah algoritma non-deterministik dimana tahap verifikasinya adalah algoritma dengan kebutuhan waktu polinom. Permasalahan dalam NP problems meliputi himpunan persoalan keputusan yang dapat diselesaikan oleh algoritma deterministik dalam waktu polinom. NPC-*problems (*NP-hard*)* merupakan permasalahan NP yang paling sukar.

Salah inti proses manajemen proyek perangkat lunak yaitu manajemen waktu pengerjaan proyek. Dalam manajemen waktu ini, penjadwalan pengerjaan proyek dapat didesain dalam diagram balok jaringan kerja dimana jalur yang mempunyai bobot terbesar dari simpul kerja awal (inisialisasi) ke simpul kerja akhir (terminasi) merupakan jalur kritis. Karena permasalahan percarian jalur dengan bobot terbesar merupakan NP-*hard* maka solusi yang dapat digunakan selain algoritma *brute-force* yaitu pendekatan algoritma *greedy* dengan solusi optimum dan teori NP.

Makalah ini membahas pemanfaatan algoritma *greedy* dan teori NP sebagai solusi alternatif pencarian jalur dengan bobot terbesar dalam diagram jaringan proyek. Aspek yang diperhatikan dalam pencarian jalur ini dalam algoritma greedy adalah cabang yang menghubungkan dua simpul dengan bobot paling besar, sedangkan dalam teori NP adalah melakukan mentransformasikan dengan permasalahan yang serupa di NP-*complete*.

***Kata kunci*—Algoritma *greedy*,teori NP,diagram jaringan kerja,jalur kritis.**

I. PENDAHULUAN

Manajemen waktu proyek merupakan proses-proses yang diperlukan untuk melengkapi proyek dengan aspek pewaktuan. Salah satu proses utama dalam manajemen waktu yaitu *activity sequencing* yang mencakup identifikasi dan pendokumentasian hubungan antar aktivitas proyek.

Salah satu output utama *activity sequencing* yaitu diagram jaringan kerja dalam metode *activity on the arrow*(AOA). Dimana simpul(*node*) menggambarkan suatu kejadian(*event*) dan panah(*arrow*) menggambarkan suatu kegiatan(activity).

Metode pencarian jalur kritis secara *brute force* atau lebih dikenal dengan *Critical Path Method* (CPM) dikembangkan pada awal tahun 1950-an oleh Morgan R. Walker of DuPont and James E. Kelley, Jr. of Remington Rand. Selain dalam proyek perangkat lunak CPM dapat diterapkan dalam berbagai macam proyek seperti konstruksi bangunan,*aerospace*,penelitian, pemeliharaan tanaman,dan sebagainya.

Diagram jaringan sangat membantu dalam memvisualisasikan rangkaian kegiatan yang penting dalam penyelesaian proyek. Diagram jaringan ini memudahkan manajer dan tim proyek dalam menyelesaikan proyek sekaligus menetukan kegiatan-kegiatan penting. Kegiatan penting adalah bagian dari rangkaian penyelesaian proyek apabila ditunda akan berdampak buruk pada keseluruhan pengerjaan proyek. Urut-urutan kegiatan demikian disebut dengan jalur kritis, sedangkan komplemen dari jalur kritis adalah *total-float* artinya apabila urut-urutan pengerjaan proyek ini ditunda tidak berdampak langsung pada penyelesaian proyek.

Keuntungan dari manajemen waktu proyek dengan menggunakan diagram jaringan yaitu :

1.)Memprediksi waktu penyelesaian setiap kegiatan dan memprediksi skala waktu untuk klien.

2.)Melihat setiap bagian yang penting untuk kemajuan seruh rencana proyek.

3.)Pemilihan team secara tepat terhadap kegiatan tertentu.

Hal penting yang harus dilakukan dalam pengguanaan CPM dalam mengkonstruksi model proyek sebagai berikut:

1.)Daftar semua aktifitas dalam penyelesaian proyek biasanya dalam bentuk *Work Breakdown Structure.*

2.)Waktu Penyelesaian tiap kegiatan.

3.)Keterhubungan kegiatan-kegiatan dalam proyek.

4.)*Logical end Points* seperti *milestones*.

Penerapan algoritma greedy dalam penentuan jalur kritis dalam diagram jaringan *activity on arrow(*AOA*)* :

1.)Pilih *initiate activity* , kegiatan yang menginisialisasi proyek selalu dipilih karena apabila ditunda berarti semua

kegiatan lain juga ditunda.
2.)Pilih *terminate activity* , sebagai simpul terminal diagram jaringan proyek.
3.)Penentuan simpul berikutnya(kejadian selanjutnya) ditentukan oleh anak panah dengan bobot terbesar.

Penyelesaian jalur dengan bobot terbesar dengan teori NP dapat diinstansiasi dari permasalahan *longest path in a directed acyclic graph* dalam sebuah makalah yang berjudul *On Approximating the Longest Path in a Graph* oleh David Karger, Rajeev Motwani dan G.D.S Ramkumar. Dalam tulisan tersebut dibuktikan bahwa penentuan jalur terpanjang dalam sebuah graf berarah non-sirkular yang diberikan titik awal dan akhir merupakan instansiasi permasalahan dari *weakly Hamiltonian graphs.* Karena bentuk graf berarah non-sirkular ini persis denga diagram jaringan network maka dapat dibuktikan bahwa permasalahan ini termasuk NP-*hard*. Karena teori NP hanya membatasi pada masalah pilihan(decision) ,maka lebih tepatnya penentuan jalur dengan bobot terbesar dari diagram jaringan proyek masuk dalam masalah NP-*hard* karena sesuai dengan definisi NP-*hard problems* adalah himpunan permasalahan NP dengan kasus optimasi, baik optimasi maksimum maupun minimum.

II. TEORI DASAR

Dalam diagram jaringan proyek simpul menggambarkan kejadian sedangkan panah menggambarkan suatu kegiatan. Dalam menggambarkan diagram jaringan harus diperhatikan hal-hal berikut ini :
1.)Setiap aktivitas hanya mewakili satu panah dalam jaringan,tidak ada sebuah aktivitas yang diwakili oleh dua anak panah dalam jaringan(tidak ada aktivitas kembar).
2.) Tidak ada 2 aktivitas yang ditunjukkan oleh 1 tail event dan head event yang sama. Situasi seperti ini dapat terjadi pada 2 atau lebih aktivitas yang dapat dilakukan secara bersama, untuk itu digunakana aktivitas dummy (dummy activity).

Jalur kritis adalah rangkaian aktivitas yang menghasilkan bobot terbesar dari simpul awal ke simpul terminal, dalam penentuan jalur kritis ini terdapat aturan sebagai berikut :

1.) Disebut aktivitas kritis bila penundaan waktu aktivitas akan mempengaruhi waktu penyelesaian keseluruhan proyek.
2.) Sedang aktivitas tidak kritis adalah jika kegiatan memiliki waktu yang dapat ditunda.
3.) Waktu yang dapat ditunda didalam aktivitas tidak kritis disebut dengan *slack* atau *float*.
4.) Jalur kritis ditunjukan oleh waktu paling lama dalam penyelesaian proyek, artinya jika ada satu saja aktivitas dijalur kritis yang tertunda, maka waktu penyelesaian proyek secara keseluruhan akan tertunda.
5.) Jalur kritis mempunyai 2 alasan:
a.)Waktu penyelesaian proyek tidak dapat dikurangi kecuali satu atau lebih aktivitas dijalur kritis dapat dipercepat penyelesaiannya b.)Penundaan aktivitas dijalur kritis akan menyebabkan penundaan waktu penyelesaian dari proyek.
6.) Penundaan di jalur tidak kritis tidak akan menunda waktu penyelesaian proyek, sejauh penundaan tidak melebihi *waktu* slack untuk setiap aktivitas tidak kritis.
7.) Penentuan jalur kritis, ada dua cara:
a. waktu terpanjang (terlama) dari setiap jalur
b. nilai 0 (null) pada perhitungan slack.

Algoritma *greedy* merupakan msalah satu algoritma penyelesaian masalah dalam ilmu komputer. Algoritma *greedy* membentuk solusi dari tiap langkahnya. Pada tiap langkah terdapat banyak pilihan yang perlu dieksplorasi. Oleh karena itu, di tiap langkah perlu diputuskan untuk mengambil pilihan yang nantinya akan menghasilkan solusi yang optimum.

Prinsip algoritma *greedy* adalah solusi optimum local(keputusan menentukan pilihan pada setiap langkahnya) akan mengarah ke sokusi optimum global. Dengan demikian, keputusan didasarkan oleh kriteria tunggal.

Sebagai algoritma penyelesaian masalah, algoritma greedy tidak selalu "menyelesaikan masalah" dengan benar. Dalam mencari solusi suatu permasalahan optimasi,*greedy* hanya bisa menjamin solusi paling optimum bila dalam setiap titik keputusan, mengambil pilihan yang paling bai(paling minimal dalam kasus mninimasi dan paling maksimal dalam kasus maksimasi) akan selalu mengarah ke tujuan akhir. Optimum global belum tentu merupakan solusi terbaik, tetapi *sub-optimum* atau *pseudo-optimum*, karena tidak semua alternative solusi akan ditinjau oleh *greedy* , hanya tiap elemen solusi dalam tiap langkah. Selain itu , hanya pemilihan fungsi pilihan yang tepat di setiap langkahnya yang akan membawa *greedy* menghasilkan solusi optimum.

Beberapa komponen dalam algoritma *greedy* adalah sebgaai berikut :

1.) Himpunan kandidat : himpunan pilihan dari mana solusi dibuat.
2.) Fungsi seleksi : memilih kandidat terbaik untuk ditambahkan ke dalam solusi.
3.) Fungsi layak : digunakan untuk menentukan apakah sebuah kandidat dapat digunakan untuk berkontribusi pada solusi.
4.) Fungsi objektif : menentukan nilai sebuah solusi.
5.) Fungsi solusi : diindikasikan saat solusi lengkap ditemukan.

NP-*hard problems*(NP-*complete*) merupakan himpunan persoalan optimasi yang dapat diselesaikan dengan algoritma non-deterministik dalam waktu polinom, dalam definisi yang lebih sederhana yaitu NP-*problems* yang paling sukar. Berbeda NP *problems* (tanpa *hard*) merupakan persoalan "keputusan".

Cara termudah untuk membuktikan suatu masalah merupakan suatu NPC adalah dengan mentranformasikan ke dalam bentuk yang serupa dengan permasalahan yang

sudah ada dalam NPC. Tidak semua permasalahan dapat ditransformasikan dalam bentuk NPC karena NPC hanya dapat menyelesaikan peroalan yang tracetable (suatu peroalan yang dapat diselesaikan dengan waktu yang reasonable baik *polynomial* maupun *non-polynomial*) . Contoh persoalan yang tidak dapat ditransformasikan dalam bentuk NPC yaitu mesin-turing yang merupkan permasalahan intracetable (permasalahan yang tidak dapat diselesaikan dengan waktu *non reasonable.*)

III. ANALISIS DAN PEMBAHASAN

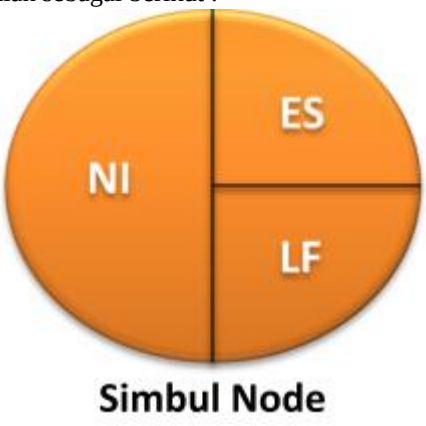
A. Perbandingan Algoritma *Brute Force* dan CPM dalam Menentukan Jalur Kritis.

Berikut adalah contoh daftar aktivitas suatu proyek sistem informasi:

No.	JOB	Waktu	Aktivitas Proyek
1	A	10	Aktivitas A,B, dan C merupakan aktivitas pertama
2	B	8	Aktivitas A mengawali aktivitas D
3	C	12	Aktivitas B mengawali aktivitas E,F,G.
4	D	22	Aktivitas C mengawali aktivitas G
5	E	27	Aktivitas D mengawali aktivitas H dan J
6	F	7	Aktivitas F mengawali aktivitas I
7	G	15	Aktivitas G mengawali aktivitas J
8	H	8	Aktivitas H,I,dan J adalah merupakan aktivitas akhir proyek.
9	I	20	-
10	J	15	-

Tabel 1- Contoh kasus manajemen waktu proyek

Dalam penggambaran diagram jaringan secara lengkap adalah sebagai berikut :



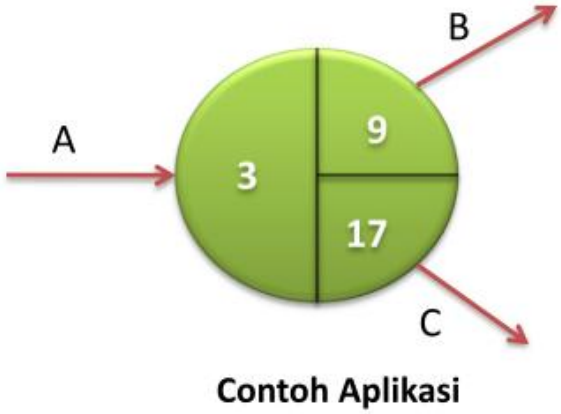
Berdasarkan aturan konstruksi diagram jaringan maka bentuk diagram jaringan sebagai berikut :

NI : Nomer identifikasi kejadian

ES : Earliest Star time (Waktu mulai tercepat) : Kapan suatu aktivitas tercepat dapat mulai dikerjakan

LS : Latest Finish time (Waktu Selesai terlama) : Kapan suatu aktivitas terlama dapat Diselesaikan

Contoh :



DARI GAMBAR SEBELAH, artinya:

1. Kejadian nomer 3
2. ES untuk aktivitas B dan C paling cepat dilakukan setelah waktu ke 9
3. LF untuk aktivitas A paling lama sampai dengan waktu ke 17

Tahap Forward Pass

(Tahap menghitung ES dari node awal maju sampai node akhir)

PERHITUNGAN :

ES1 = 0, karena start event

ES2 = ES1 + W(A) = 0 + 9 = 9

ES3 = ES1 + W(B) = 0 + 10 = 10

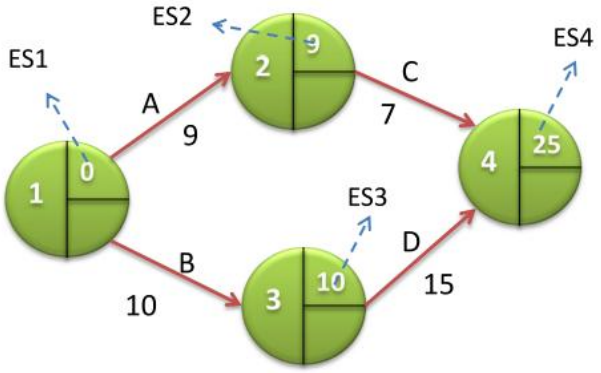
ES4 = ES2 + W(C) = 9 + 7 = 16

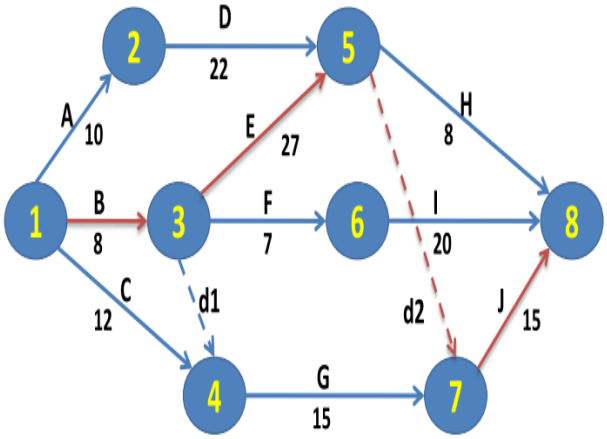
ES3 + W(D) = 10 + 15 = 25

ES4 yang diambil terbesar nilainya yaitu 25

Catatan:

W(A): Waktu aktivitas A





Gambar 1-Diagram jaringan dengan d1,d2 sebagai aktivitas dummy.

Dengan algoritma brute-force maka solusi yang mungkin adalah :

1. Jalur A-D-H = 10 + 22 + 8 = 40
2. Jalur B-E-H = 8 + 27 + 8 = 43
3. Jalur B-E-d2-J = 8 + 27 + 0 + 15 = 50
4. Jalur B-F-I = 8+7+20 =35
5. Jalur B,d1,G,J = 8 + 0 + 15 +15 = 38
6. Jalur C,G,J = 12 + 15 +15 = 42

Berdasarkan metode *brute force* tersebut maka didapatkan jalur kritis adalah B,E,J. Ini merupakan solusi yang paling baik namun kurang efektif.

Sedangkan dalam metode CPM dibagi menjadi dua *task* yaitu :

a. Tahap Forward Pass, untuk menghitung ES
b. Tahap Backward Pass, untuk menghitung LF

Berikut adalah perhitungan ES pada tiap simpul :

JOB	Kejadian	Rumus Pada Node	Perhitungan	ES
A	Awal Kejadian	ES1(A)=0	-	0
B	Awal kejadian	ES1(B)=0	-	0
C	Awal Kejadian	ES1(C)=0	-	0
D	Setelah Pekerjaan A	ES2(D)=ES1(A)+W(A)	ES2=0+10= 10	10
E	Setelah Pekerjaan B	ES3(E)=ES1(B)+W(B)	ES3=0+8=8	8
F	Setelah Pekerjaan B	ES3(F)=ES1(B)+W(B)	ES3=0+8=8	8
G	Setelah Pekerjaan dummy1 dan C	ES4(G)=W(B)+W(d1) ES4(G)=ES1(C)+W(C)	ES3=0+8=8	12
H	Setelah Pekerjaan D dan E	ES5(H)=ES2(D)+W(D) ES5(H)=ES3(E)+W(E)	ES5=10+22= 32 ES5=8+27= 35	35
I	Setelah Pekerjaan F	ES6(I)=ES3(F)+W(F)	ES6=8+7=15	15
J	Setelah Pekerjaan dummy2 dan G	ES7(J)=ES5+W(d2) ES7(J)=ES4(G)+W(G)	ES7=35+0=3 5 ES7=12+15= 27	35

Untuk menentukan jalur kritis dengan metode CPM dengan mencari *slack* atau *float* yang bobotnya sama dengan 0. Untuk menetukan *slack* harus ditententukan terlebih dahulu LF dan ES dari diagram jaringan. ES untuk tiap-tiap aktifitas telah didapat dari tabel perhitungn di atas sedangkan untuk mencari LF yaitu dengan cara *backward pass*. Harus ditentukan nilai ES paling tinggi pada diagram jaringan untuk mencari nilai LF dari aktivitas yang mendahuluinya dengan cara mengurangkan nilai LF terhadap nilai penyelesaian aktivitas tersebut.

JOB	Waktu	ES	LF	LS	EF	Slack
A	10	0	13	3	10	3
B	8	0	8	0	8	0
C	12	0	20	8	12	8
D	22	10	35	13	32	3
E	27	8	35	8	35	0
F	7	8	30	23	15	15
G	15	12	35	20	27	8
H	8	35	50	42	43	7
I	20	15	50	30	35	15
J	15	35	50	35	50	0
Finish	0	50	50	50	50	0

Tabel 3-Penetuan slack

Ket : LS = LF-W
EF = ES + W
Slack = LS – ES

Jika dibandingkan dengan algoritma brute-force sebelumnya dengan mengiterasi semua kemungkinan solusi, membandingkan semua solusi untuk mencapai

sousi optimum menghasilkan solusi yang sama yaitu jalur kritis dibentuk dari simpul-simpul B,E,J.

B. Algoritma *Greedy* untuk Penentuan Jalur Kritis.

Dalalm menentukan lintasan terpanjang dengan algoritma greedy seperti biasa dimulai dengan mencari solusi optimum local kemudian dilanjutkan ke solusi optimum global. Berdasarkan graf dalam gambar 1, maka untuk mencari solusi berdasarkan dengan sifat greedy yang pertama adalah menetukan simpul tujuan dengan nilai bobot anak panah paling tinggi. Setelah menuju simpul tersebut ,untuk mencapai simpul berikutnya dengan cara ang sama yaitu mencari simpul dengan anak panah penghubung nilainya paling tinggi. Langkah ini dilanjutkan hingga menuju simpul terminal.

Dapat dilihat dalam gambar bahwa dari simpul awal ke simpul berikutnya jalur dengan bobot maksimum yaitu simpul 4, dengan bobot 12 yaitu aktivitas C. setelah dari simpul 4 ditentukan solusi berikutnya dengan mencari solusi maksimum berikutnya yaitu node 5 dengan bobot 22, dari simpul 5 selanjutnya memeilih simpul berikutnya yaitu simpul 7 dengan bobot sebesar 15. Jadi total bobot maksimum yang diperoleh dengan algoritma greedy yaitu 42.

Yang membuat rancu sehingga permasalahan ini termasuk dalam NP-*Complete* adalah dalam representasi matriks untuk graf yang tak terhubung diberi nilai tak hingga sehingga pada saat kita memilih jalur dengan bobot terbesar dengan hanya melihat representasi dalam bentuk matriks untuk graf berarah tidak lengkap maka kita akan selalu mendpatkan nilai tak hingga.

Berikut adalah properties penyelesaian dari algoritma *greedy* yaitu

- Himpunan Kandidat : himpunan jalur dengan bobot tertentu.
- Himpunan Solusi : himpunan solusi penentuan jalur dengan bobot terbesar.
- Fungsi Seleksi : memilih anak panah dengan nilai tertinggi.
- Fungsi Layak : memeriksa apakah jalur dalam lingkup himpunan kandidat.
- Fungsi Obyektif : jumlah anak panah yang minimal agar , jumlah aktivitas paling kecil paling minimal.

Berdasarkan teorema 3 dalam makalah yang berjudul *An Approximation to Find Longest Path* , dalam kasus penentuan jarak terpanjang masih merupakana masalah *tracetable* sehingga kita masih bisa menentukan waktu yang reasonable dalam penyelesaian masalah ini.

Berikut adalah algoritma weakly – Hamiltonian dalam penentuan suatu graf berarah :

Algorithm LONG-PATH:
Input: 1-tough graph G(V ; E).
Output: Long path P, where P is an ordered list of vertices.

1. Pick an arbitrary vertex v 2 V ;
2. P ADD-PATH ((G(V ; E); v);
3. return P

Procedure ADD-PATH:

Input: 1-tough graph G(V ; E) and vertex v 2 V
Output: Path P.

1. if jV j = 1 then P v else begin

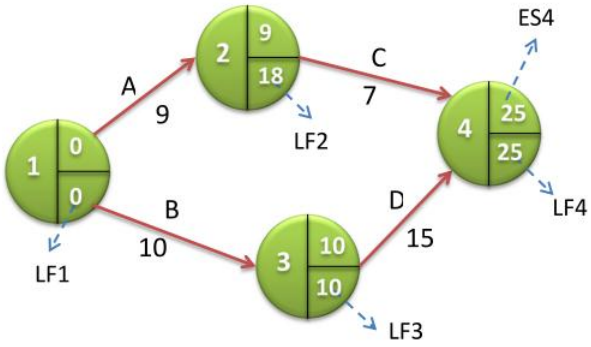
Compute the connected components of $G[V - \{v\}]$;
Let W be the largest component $G[V - \{v\}]$;
Pick an arbitrary neighbor w of v from W;
$P \leftarrow v \circ$ ADD-PATH $(G[W], w)$;

2. end
return P.

dalam prosedur tersebut terdapat instruksi yang pertama adalah memilih anak panah dengan julah paling besar kemudian bentuk graf baru , dengan simpul awal tertentu sehingga algoritma ini selalu menghasilkan kompleksitas ang lebih efisien jika dibandigkan brute-force.

Tahap Backward Pass
(Tahap menghitung LF dari node akhir mndur sampai node awal)

P E R H I T U N G A N :
LF4 = ES4, yaitu : 25
LF3 = LF4 - W(D) = 25 – 15 = 10
LF2 = LF4 - W(C) = 25 – 7 = 18
LF1 = LF3 - W(B) = 10 – 10 = 0
LF2 - W(A) = 18 – 9 = 9
LF1 yang diambil terkecil nilainya
yaitu



P E R H I T U N G A N :
LF4 = ES4, yaitu : 25
LF3 = LF4 - W(D) = 25 – 15 = 10
LF2 = LF4 - W(C) = 25 – 7 = 18
LF1 = LF3 - W(B) = 10 – 10 = 0
LF2 - W(A) = 18 – 9 = 9
LF1 yang diambil terkecil nilainya

IV. KESIMPULAN

Dari pembahasan di atas dapat dihasilkan kesimpulan
sebagai berikut

1. Algoritma greedy dapat digunakan pada penyelesaian permasalahan optimasi yang melibatkan banyak kegiatan memilih pada setiap tahapnya.
2. Untuk menentukan strategi greedy yang baik, fungsi seleksi yang dibuat harus dipikirkan baik-baik berdasarkan apa yang perlu didahulukan dan batasan-batasan pemilihan, karena yang membedakan kebaikan hasil suatu algoritma greedy terletak pada pendefinisian fungsi seleksinya.
3.Algoritma Brute Force menghasilkan solusi yang sama dengan CMP
4. masalah pencarian jalur terpanjang merupakan NP-*complete*.

REFERENSI

[1] Munir, Rinaldi. Strategi Algoritma. 2009. Bandung: Penerbit ITB, halaman 26-32

[2] _ "Algorithms/Greedy Algorithms". Sumber http://en.wikibooks.org/wiki/Algorithms/Greedy_Algorithms

[3]http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.54.2580&rep=rep1&type=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, 20 Desember 2013

Jais Anasrulloh Ja'fari | 13511036