Penggunaan Metode *Dynamic Programming* Dalam Perencanaan dan Pengendalian Proyek dengan PERT/CPM

Reza Rahman Mohammad[1], M. Randy Desmond Ibrahim[2], Eko Budhi Susanto[3]

Laboratorium Ilmu dan Rekayasa Komputasi
Departemen Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail :
if14061@students.if.itb.ac.id[1], if14069@students.if.itb.ac.id[2],
if14075@students.if.itb.ac.id[3]

Abstrak

PERT/CPM atau dikenal dengan PERT-*type system* adalah sebuah prosedur perencanaan, penjadwalan, dan pengorganisasian proyek-proyek berskala besar yang didasarkan atas penggunaan jaringan dan teknik-teknik jaringan. Makalah ini mempresentasikan penggunaan metode dynamic programming untuk menentukan jalur kritis dalam perhitungan CPM(*Critical Path Method*) yang digunakan dalam PERT-*type system*.

***Kata kunci**: Dynamic Programming,* CPM, jalur kritis.

1. Pendahuluan

PERT-*type system* menggunakan *network* (jaringan kerja) untuk menggambarkan inter-relasi di antara elemen-elemen proyek. Setelah network suatu proyek dapat digambarkan, langkah berikutnya adalah mengestimasi waktu yang diperlukan untuk masing-masing aktivitas, dan menganalisis seluruh diagram jaringan untuk menentukan waktu terjadinya masing-masing kejadian (*event*). Dalam mengestimasi dan menganalisis waktu ini, akan kita dapatkan satu atau beberapa lintasan tertentu dari kegiatan-kegiatan pada network tersebut yang menentukan jangka waktu penyelesaian seluruh proyek. Lintasan ini disebut *lintasan kritis (critical path)*. Selain itu ada pula lintasan yang tidak kritis yang mempunyai waktu untuk bisa terlambat, yang dinamakan *float*. Setiap jaringan memiliki titik inisiasi sebagai awal dan titik terminasi sebagai tanda berakhirnya suatu jaringan proyek.

Adapun tujuan dari PERT-*type system* ini antara lain:

1. menentukan total waktu untuk menyelesaikan satu proyek apabila tidak ada delay yang terjadi.
2. menentukan kapan setiap aktivitas (*node*) paling lambat harus dimulai dan berakhir untuk memenuhi waktu proyek yang telah ditentukan (*Latest Start* dan *Latest Finish*).
3. menentukan kapan setiap aktivitas (*node*) paling cepat harus dimulai dan berakhir untuk memenuhi waktu proyek yang telah ditentukan (*Early Start* dan *Early Finish*).
4. menentukan mana aktivitas yang tidak punya waktu delay (*critical bottleneck*)
5. berapa lama delay yang bisa ditoleransi dalam penyelesaian suatu proyek.

Dynamic Programming adalah metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan langkah (*step*) atau tahapan (*stage*) sedemikian sehingga solusi dari persoalan dapat dipandang dari serangkaian keputusan yang saling berkaitan.

Metode *Dynamic Programming* dianggap sesuai untuk digunakan pada PERT-*type system* karena keduanya memiliki beberapa kriteria yang serupa dalam penyelesaian masalah, antara lain:

- Proyek yang diproses hanya memiliki satu *initial event* dan satu *terminal event.*
- Solusi pada setiap tahap dibangun dari hasil solusi tahap sebelumnya.
- Terdapat sejumlah berhingga pilihan yang mungkin dalam membentuk jalur pada sebuah jaring proyek (*Project network*).
- Cara perhitungan dilakukan harus dengan 2 cara, yaitu perhitungan maju (forward computation) dan perhitungan mundur (backward computation).

2. Ruang Lingkup

PERT-*type system* adalah sebuah prosedur gabungan dari dua prosedur utama diantara prosedur-prosedur perencanaan dan pengendalian proyek. Dua prosedur tersebut dikenal sebagai PERT(*Program Evaluation*

and Review Technique) dan CPM(*Critical Path Method*).

Makalah ini mempresentasikan metode *dynamic programming* untuk menentukan jalur kritis dalam perhitungan CPM.

Perhitungan yang dapat dilakukan dengan *dynamic programming* antara lain:
1. menentukan total waktu untuk menyelesaikan satu proyek apabila tidak ada *delay* yang terjadi.
2. menentukan mana aktivitas yang tidak punya waktu *delay* (*critical bottleneck*)

3. Manajemen Proyek dengan PERT-*type System*

Proses Manajemen Proyek bertujuan untuk mengoptimalkan proses pengerjaan suatu proyek. Hal-hal yang dapat diperhitungkan untuk membantu manajemen proyek antara lain:
1. Jalur kritis(*Critical Path*)
2. ES=*Early Start*, dan EF=*Early Finish*
3. LS=*Latest Start*, dan L=*Latest Finish*
4.*Delay*

3.1. Membangun Jaringan

Untuk memulai manajemen proyek dengan dengan PERT-*type system* pertama-tama kita menerima masukan berupa proses kerja yang berbentuk graf berarah.

Setiap proses kerja kita anggap sebagai simpul. Setiap simpul memiliki nama dan durasi. Sisi menghubungkan setiap proses kerja ke proses kerja selanjutnya. Sebuah jaringan proyek memiliki awal(*Start*) dan akhir(*Finish*) proyek. Simpul Start menjadi tempat bermulanya proses kerja, sedangkan simpul Finish tempat terminasi proses kerja. Jadi semua proses kerja pertama terhubung dengan Start dan proses kerja terakhir terhubung dengan finish.

3.2. Mencari Jalur Kritis

Setelah jaringan terbentuk, selanjutnya kita mencari jalur kritis. Jalur kritis adalah jalur dari kegiatan-kegiatan pada jaringan tersebut yang menentukan jangka waktu penyelesaian seluruh proyek, yaitu jalur dengan total waktu maksimum. Jalur kritis ini diperlukan dalam pengestimasian penganalisisan waktu untuk mengoptimalkan proses kerja proyek.

3.2. Perhitungan Maju

Perhitungan maju dimulai dari *initial event*(simpul *Start*) menuju *terminal event*(simpul *finish*). Maksudnya adalah untuk menghitung saat paling cepat dimulainya serta diselesaikannya aktivitas-aktivitas (ES=*Early Start*, dan EF=*Early Finish*).

3.3. Perhitungan Mundur

Perhitungan mundur dimulai dari *terminal event* menuju ke *initial event*. Tujuannya untuk menghitung saat paling lambat saat terjadinya dimulainya dan diselesaikannya aktivitas (LS=*Latest Start*, dan LF=*Latest Finish*).

3.4. Perhitungan Keterlambatan

Perhitungan keterlambatan untuk mengetahui toleransi keterlambatan setiap proses (*Delay*). Dihitung dengan cara mengurangi LF dengan EF atau LS dengan ES pada setiap proses.

Delay suatu proses dalam jalur kritis adalah nol. Hal ini menyebabkan, jika terjadi keterlambatan waktu proses dapat mengakibatkan keterlambatan penyelesaian proyek.

Jadi batas keterlambatan suatu proses tidak boleh lebih besar dari *Delay*-nya.

4. Penerapan *Dynamic Programming*

Setiap simpul dari jaringan proses kerja memiliki durasi(d). Durasi penyelesaian kerja adalah durasi maksimum(d_{max}) untuk seluruh proses kerja.

Jalur kritis adalah jalur yang menghasilkan dmax. Jalur yang memiliki *delay* nol. *Dynamic Programming* dalam persoalan ini diterapkan dalam pencarian jalur kritis.

Penerapan metode *Dynamic Programming* dalam masalah ini secara umum dapat dituliskan sebagai berikut:

$$f(s) = d_s \qquad \text{(basis)}$$

$$f(s) = \max_{i=1}^{ks}\{d_s + f(next_i(s))\} \qquad \text{(rekurens)}$$

keterangan :
s : simpul proses kerja
d : durasi kerja
ks : jumlah anak pada simpul s
$next_i$: simpul selanjutnya ke-i, merepresentasikan anak s yang ke-i

Algoritmanya dalam bentuk pseudo code adalah sebagai berikut:

```
function dpCPM(L: Jaringan; A: simpul): real;
var
  max, hitung : real;
  temp: simpul;
begin
  if (A tidak memiliki anak) then
{basis}
    max:= A.durasi
  else
{rekurens}
  begin
   for (temp:= semua anak A) do
   begin
    hitung:=A.durasi+dpCPM(L, temp);
    if hitung>max then
      max:= hitung;
   endfor;
  endif;
  dpCPM := max;
end; {end function}
```

Basis adalah simpul yang tidak memiliki anak (jumlah anak nol). Anak disini maksudnya proses setelah proses pada simpul yang bersangkutan, yaitu simpul yang ditunjuk oleh sisi dari simpul lain.

Jika ingin mendapat waktu total maksimum dari sebuah proses jaringan kerja kita dapat menggunakan algoritma diatas dengan masukan sebuah jaringan dan simpul *Start* jaringan tersebut, contoh:

```
dpCPM(L, getStart(L));
```

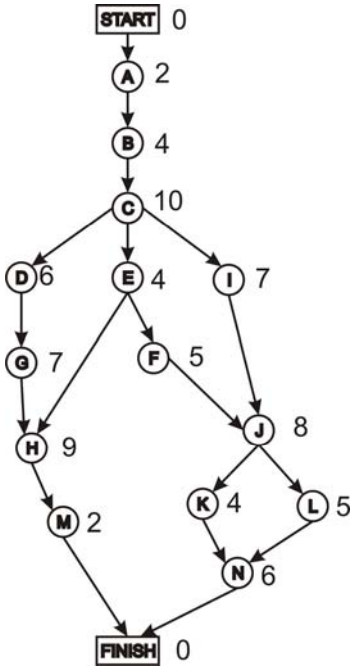
dengan L adalah sebuah jaringan, dan getStart adalah fungsi yang mengembalikan sebuah simpul *Start* pada jaringan.

5. Studi Kasus

Sebuah Perusahaan konstruksi mendapat suatu proyek dengan *waktu* pengerjaan maksimum 47 minggu. Aktivitas-aktivitas yang harus diselesaikan untuk menyelesaikan proyek tersebut adalah sebagai berikut:

Aktivitas	Deskripsi Aktivitas	Proses sebelum	Perkiraan durasi
A	Menggali	–	2 minggu
B	Membangun pondasi	A	4 minggu
C	Membangun Kerangka	B	10 minggu
D	Membangun kuda-kuda	C	6 minggu
E	Pasang pipa air bag. luar	C	4 minggu
F	Pasang pipa air bag. dalam	E	5 minggu
G	Membangun tembok	D	7 minggu
H	Cat dinding bagian luar	E, G	9 minggu
I	Instalasi listrik	C	7 minggu
J	Pasang papan dinding	F, I	8 minggu
K	Pasang ubin	J	4 minggu
L	Cat dinding bagian dalam	J	5 minggu
M	Instalasi perabot bag. luar	H	2 minggu
N	Instalasi perabot bag. dalam	K, L	6 minggu

Dari rangkaian proses diatas dapat dibentuk sebuah jaringan proyek seperti dibawah ini:



5.1. Pencarian Jalur Kritis Dengan Metode *Brute Force*

Dengan metode *Brute Force* kita mencoba setiap kemungkinan satu persatu.
Macam-macam jalur pada jaringan proyek diatas:

Start – A – B – C – D – G – H – M – Finish (40)
Start – A – B – C – E – H – M – Finish (31)
Start – A – B – C – E – F – J – K – N – Finish (43)
Start – A – B – C – E – F – J – L – N – Finish (44)
Start – A – B – C – I – J – K – N Finish (41)
Start – A – B – C – I – J – L – N Finish (42)

Jalur kritis:

Start – A – B – C – E – F – J – L – N – Finish

Dengan total waktu maksimum untuk proyek tersebut, yaitu 44 minggu.

5.2. Pencarian Jalur Kritis Dengan *Dynamic Programming*

Dengan menggunakan metode *dynamic programming* persoalan ini dapat diselesaikan dengan cara sebagai berikut:

$$f(A) = \max_{i=1}^{1}\{d_A + f(next_i(A))\}$$

Yang artinya mengembalikan nilai maksimum dari durasi simpul A ditambah dengan jumlah durasi maksimum simpul-simpul yang bertetangga dengan A.

Simpul N :

I	d_N	$f(next_i(N))$	f
1(Finish)	6	0	6 + 0

Simpul L :

I	d_L	$f(next_i(L))$	f
1(N)	5	$6 + f(next_i(N))$	5 + 6

Simpul J :

I	d_J	$f(next_i(J))$	f
1(K)	8	$4 + f(next_i(K))$	8 + 10
2(L)	8	$5 + f(next_i(L))$	8 + 11

Simpul F :

I	d_F	$f(next_i(F))$	f
1(J)	5	$8 + f(next_i(J))$	5 + 19

Simpul E :

I	d_E	$f(next_i(E))$	f
1(F)	4	$5 + f(next_i(F))$	4 + 24

Simpul C :

I	d_C	$f(next_i(C))$	f
1(D)	10	$6 + f(next_i(D))$	10 + 24
2(E)	10	$4 + f(next_i(E))$	10 + 28
3(I)	10	$7 + f(next_i(I))$	10 + 26

Dalam persoalan ini simpul A-B-C sudah pasti mengembalikan nilai yang sama, jadi bisa kita tulis:

Simpul A :

i	d_A	$f(next_i(A))$	f
1(B - C)	2	$2 + 4 + 10 + f(next_i(F))$	2 + 4 + 42

Dari tabel diatas didapat solusi untuk persoalan ini:

Start – A – B – C – E – F – J – L – N – Finish

0 + 2 + 4 + 10 + 4 + 5 + 8 + 5 + 6 + 0 = 44 minggu

Setelah waktu maksimum dan jalur kritis ditemukan, proses manajemen masuk ke tahap berikutnya.

6. Kesimpulan

Penggunaan *Dynamic Programming* dalam pencarian jalur kritis dan waktu maksimum disini dimaksudkan untuk mempermudah proses perhitungan CPM yang sudah ada. Karena untuk melakukan pencarian jalur kritis dengan metode brute force biasa akan sangat memakan waktu untuk masukan sebuah jaringan proses kerja yang besar.

Selain pencarian jalur kritis dan waktu maksimum, masih banyak lagi yang harus diperhitungkan dalam perencanaan dan pengendalian proyek dengan PERT-*type system*.

Dari metode CPM sendiri hal-hal yang tidak dibahas antara lain ES, EF, LS, LF yang berguna untuk menghitung *delay* setiap proses.

Hal-hal itu semua diatas belum termasuk metode PERT, yaitu metode pencarian jalur kritis dan waktu maksimum dengan tambahan *input* durasi optimis dan pesimis, dan melakukan perhitungan dengan probabilitas.

7. Referensi

1. M. Rinaldi, *Diktat Kuliah IF 2251 Strategi Algoritmi*k, Institut Teknologi Bandung, Januari 2005.
2. D. Ahmad & Tjutju Tarliah Dimyati, *Operations Research ; Model-model pengambilan keputusan*, Sinar Baru Algensindo, 2004.
3. Hieberman, Hillier, *Introduction to Operation Research Eighth Edition*, McGraw-Hill International Edition, 2005.
4. Hieberman, Hillier, *Operation Research For Engineering*, McGraw-Hill International Edition, 2005.