C.   PROGRAM DINAMIK (KONTEN IV)

Deskripsi Singkat

Dynamic programming adalah strategi untuk membangun masalah optimal bertingkat, yaitu masalah yang dapat digambarkan dalam bentuk serangkaian tahapan (*stage*) yang saling mempengaruhi. Umumnya tiap tahapan mempunyai 4 (empat) variabel yang mempunyai pengaruh, baik langsung maupun tidak langsung terhadap tahapan lainnya dari sistem.

Input untuk tahapan n, X_n, yang tergantung dari keputusan yang dibuat pada tahapan terdahulu atau tergantung dari input asal yang tetap pada sistem.

1. Set keputusan pada tahap n, D_n yang menentukan kondisi atau syarat operasi dari tahapan.
2. Output dari tahapan n, X_{n-1} yang biasa tergantung dari input pada tahapan n dan keputusan D_n.
3. Hasil dari tahapan n, R_n yang merupakan ukuran bagi konstribusi tahapan n terhadap fungsi tujuan sistem keseluruhan (ongkos, keuntungan, manfaat atau ukuran lain). Biasanya hasil ini merupakan gambaran dari suatu tahapan n dan output pada tahapan n.

Pemrograman dinamis merupakan suatu teknik matematis untuk pembuatan serangkaian keputusan yang saling berhubungan. Teknik tersebut menyediakan prosedur sistematis untuk menentukan kombinasi keputusan yang optimal. (Hiller, Frederick S, 1994)

Berbeda dengan program linear, tidak terdapat rumusan matematis standar dalam masalah pemrograman dinamis. Akan tetapi pemrograman dinamis adalah suatu tipe pendekatan umum dalam pemecahan masalah dan persamaan-persamaan tertentu yang digunakan harus dibuat sesuai dengan situasi yang sifatnya individual. Dengan demikian sedikit kecerdikan dan pengetahuan tentang struktur umum masalah pemrograman dinamis diperlukan untuk mengenali kapan suatu masalah dapat dipecahkan dengan prosedur pemrograman dinamis dan bagaimana menyelesaikannya. Kemampuan ini dapat dikembangkan dengan mengenali berbagai macam aplikasi pemrograman dinamis dan mempelajari ciri-ciri yang sama dari keadaan-keadaan tersebut. Dalam Program Dinamik akan dibahas Model Program Dinamik (PD),
Unsur dari model PD, Representasi bersama yang berulang, Program Deterministik (Perhitungan maju / mundur) dan Program Probabilistik (Problem dari ukuran PD)

Kompetensi yang Dapat Dicapai Oleh Mahasiswa

1. Mampu membuat Model Program Dinamik
2. Mampu menganalisis Program Deterministik dengan langkah maju dan mundur
3. Mampu menganalisis
4. Program Probabilistik
5. Mampu membuat tugas Tim, dan Mandiri serta dapat mempresentasikannya dengan informatif dan jelas

1. Program Dinamik

Pemrograman dinamis (*Dynamic programing)* adalah **teknik matematika yang dirancang khusus untuk meningkatkan efisiensi komputasi, terutama pada beberapa masalah komputasi**. Ide dasar dari teknik ini adalah untuk menguraikan masalah besar, menjadi beberapa submasalah, yang mana secara komputasi lebih mudah untuk dilakukan pengaturan. (Hamdy, A. Taha,1996)

Program dinamis ini dirancang untuk mengatasi kesulitan sebagai berikut :

a. Masalah diuraikan menjadi sub-sub masalah yang disebut tahapan dan setiap sub masalah yang akan melalui alternatifnya saja, sehingga tidak perlu untuk menghitung semua kombinasi di awal.

b. Karena optimisasi dilakukan untuk setiap sub masaalah, semua kombinasi yang tidak optimal secara sistematis akan dibuang.

c. Semua sub masalah dihubungkan bersama dalam jalur khusus, sehingga tidak memungkinkan untuk mengoptimalkan lebih banyak kombinasi.

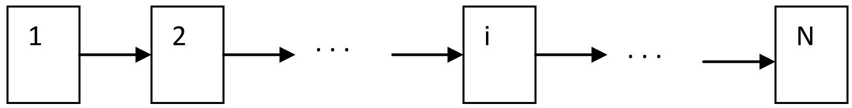
Poin-poin di atas menerjemahkan formula program dinamis dalam tiga unsur dasar Stage (tahapan)

0. Alternatif (variabel keputusan) pada setiap tahapan dan fungsi terkait.

1. Keadaan sistem pada setiap tahapan.

Stage (tahapan) mewakili sebuah bagian dari masalah untuk membuat keputusan. Sebagai contoh, penganggaran modal, solusi masalah tersebut adalah dengan memilih satu alternatif untuk setiap pabrik. Dengan demikian, pabrik tersebut menggambarkan tahapan yang terjadi. Hal itu akan menjadi suatu hal yang tidak lain untuk diperhatikan sebagai tahapan yang terdiri atas alternatif dari berbagai pabrik. Keterkaitan antara alternatif tidak saling ekslusif, oleh sebab itu kita tidak bisa membandingkannya satu sama lain.

Ciri khas model program dinamis dengan N tahapan, dapat digambarkan dengan grafik seperti pada gambar berikut.



Proses optimalisasi, normalnya dimulai dari tahap I dan maju dengan sukses, tahapan demi tahapan, hingga sampai N, merupakan optimalisasi. Determinasi dari alternatif di dalam setiap tahapan adalah bagian integral dari definisi tahapan, sehingga siap diidentifikasi. Keterkaitan dari setiap tahapan adalah fungsi balik dari variabel keputusan dalam mengevaluasi setiap alternatif. Dalam masalah penganggaran modal R_i, m_i adalah hasil dari alternatif m_i pada tahapan (pabrik) i.

Keadaan sistem mungkin adalah yang paling penting dalam konsep model pemrograman dinamis. Ini merupakan **"*Link*'** antartahapan, sehingga bila setiap tahap dioptimalkan secara terpisah, maka keputusan yang dihasilkan secara otomatis layak untuk seluruh masalah. Selain itu, memungkinkan seseorang untuk membuat keputusan yang optimal untuk membuat keputusan yang optimal untuk sisa tahapan tanpa harus memeriksa efek dari masa depan keputusan yang dibuat sebelumnya.

2. Representasi Persamaan Yang Berulang

Pada program dinamis, rangkaian keputusan yang optimal dibuat dengan menggunakan **Prinsip Optimalitas**. Prinsip Optimalitas adalah ***jika solusi total optimal, maka bagian solusi sampai tahap ke-k juga optimal*.**

Prinsip optimalitas berarti bahwa jika kita bekerja dari tahap k ke tahap k+1, kita dapat menggunakan hasil optimal dari tahap k tanpa harus kembali ke tahap awal.

ongkos pada tahap k +1 = (ongkos yang dihasilkan pada tahap k) +
(ongkos dari tahap k ke tahap k + 1)

Dengan prinsip optimalitas ini dijamin bahwa pengambilan keputusan pada suatu tahap adalah keputusan yang benar untuk tahap-tahap selanjutnya.

Keistimewaan dasar yang mencirikan masalah pemrograman dinamis diberikan dan didiskusikan berikut ini.

a. Permasalahan dapat dibagi dalam tahap-tahap, dengan suatu keputusan kebijakan (*policy decision*) diperlukan di setiap tahap.

Masalah perjalanan kereta dibagi dalam empat tahap yang berhubungan dengan empat cabang perjalanan. Keputusan kebijakan pada setiap tahap adalah asuransi asuransi yang harus dipilih (yaitu tujuan mana yang harus dipilih pada perjalanan berikutnya). Dengan cara yang sama, masalah pemrograman dinamis yang lain memerlukan pembuatan suatu urutan keputusan yang saling berhubungan, dimana setiap keputusan berhubungan dengan suatu tahap permasalahan.

b. Setiap tahap memiliki sejumlah keadaan (states) yang bersesuaian.

Keadaan yang bersesuaian dengan setiap tahap pada masalah perjalanan kereta adalah keadaan (atau daerah) dimana si pencari keberuntungan berada ketika berangkat pada cabang perjalanan tertentu. Secara umum, keadaan adalah berbagai kondisi yang mungkin, di mana system berada pada tahap tertentu dari keseluruhan permasalahan. Banyaknya keadaan mungkin saja terbatas (seperti pada masalah perjalanan kereta) ataupun tak terbatas.

c. Pengaruh keputusan kebijakan pada setiap tahap adalah untuk merubah keadaan sekarang menjadi keadaan yang berkaitan dengan tahap berikutnya (menurut suatu distribusi peluang tertentu).

Keputusan si pencari keberuntungan tentang tujuan berikutnya membawanya dari keadaan sekarang kepada keadaan berikutnya. Setiap simpul berpadanan dengan keadaan. Jaringan kerja tersebut berisi kolom-kolom yang memuat node, di mana setiap kolom berpadanan dengan tahap, sehingga aliran dari suatu simpul hanya dapat pergi ke simpul dalam kolom berikutnya di sebelah kanan. Hubungan dari suatu simpul ke simpul dalam kolom berikut berpadanan dengan keputusan kebijakan yang mungkin tentang keadaan mana yang harus dituju. Nilai-nilai yang diberikan pada setiap kaitan biasanya diinterpretasikan sebagai kontribusi sekarang pada fungsi tujuan dari pembuatan keputusan kebijakan tersebut. Umumnya tujuan tersebut berhubungan dengan penemuan rute terpendek atau terpanjang dalam jaringan kerja.

d. Prosedur penyelesaian dirancang untuk menemukan suatu kebijakan optimal pada keseluruhan masalah, yaitu pemberian keputusan kebijakan optimal dari setiap tahap untuk setiap kemungkinan keadaan.

Untuk masalah perjalanan kereta, prosedur penyelesaian dilakukan dengan menyusun suatu table untuk setiap tahap(n) yang memberi keputusan optimal (X_n*) untuk setiap keadaan (s) yang mungkin. Jadi sebagai tambahan pada identifikasi ketiga penyelesaian optimal (rute optimal) untuk keseluruhan masalah, hasil tersebut juga menunjukkan kepada si pencari keberuntungan bagaimana ia harus menlanjutkan perjalanan bila ia menuju kesuatu keadaan yang bukan merupakan rute optimal. Untuk setiap masalah program, pemrograman dinamis menyediakan petunjuk kebijakan apa yang harus dilakukan apa setiap keadaan (yaitu mengapa keputusan tertentu diambil setelah keadaan tertentu pada tahap tertentu hal ini dikenal sebagai keputusan kebijakkan). Penyediaan informasi tambahan ini disamping memberikan penyelesaian optimal (urutan keputusan optimal) juga dapat membantu dengan berbagai jalan, termasuk analisa kepekaan.

e. Bila diketahui keadaan sekarang, kebijakkan optimal untuk tahap-tahap yang tersisa adalah bebas terhadap kebijakkan yang dipakai pada tahap-tahap sebelumnya. (ini adalah prinsip ke optimalan dalam pemrograman dinamis).

Misal diketahui keadaan dimana si pencari keberuntungan berada sekarang, asuransi asuransi optimal (dan rute yang bersesuaian) dari titik ini adalah bebas terhadap cara ia sampai ke sana. Untuk masalah pemrograman dinamis secara umum, pengetahuan tentang keadaan system sekarang yang membawa semua informasi tentang tingkah laku sebelumnya menjadi perlu untuk menentukan kebijakan optimal. Setiap masalah yang tidak memenuhi sifat ini tidak dapat dirumuskan sebagai masalah pemrograman dinamis.

f. Prosedur penyelesaian dimulai dengan menemukan kebijakan optimal untuk tahap terakhir.

Kebijakan optimal untuk tahap terakhir memberikan keputusan kebijakan yang optimal untuk setiap kemungkinan keadaan pada tahap tersebut. Penyelesaian pada masalah satu tahap ini biasanya trivial, seperti pada masalah perjalanan kereta.

g. Tersedia hubungan rekursif yang mengindentifikasi kebijakan optimal pada tahap n, bila diketahui kebijakan optimal untuk tahap (n+1).

Suatu hubungan rekursif digunakan untuk menghubungkan kebijaksanaan optimum pada tahap *n* dengan *n*-1.

Ada dua macam prosedur rekursif yaitu :

- *Forward recursive equation* (perhitungan dari depan ke belakang)
- *Backward recursive equation* (perhitungan dari belakang ke depan)

Misalkan x_1, x_2, …, x_n menyatakan peubah (*variable*) keputusan yang harus dibuat masing-masing untuk tahap 1, 2, …, *n*. Maka,

- ***Forward recursive*** *equation* (perhitungan dari depan ke belakang). Program dinamis bergerak mulai dari tahap 1, terus maju ke tahap 2, 3, dan seterusnya sampai tahap *n*. Runtunan peubah keputusan adalah x_1, x_2, …, x_n.
- ***Backward recursive*** *equation* (perhitungan dari belakang ke depan). Program dinamis bergerak mulai dari tahap *n*, terus mundur ke tahap $n – 1$, $n – 2$, dan seterusnya sampai tahap pertama Runtunan peubah keputusan adalah x_n, x_{n-1}, …, x_1.
-

Adapun Langkah-langkah Pengembangan Algoritma Program Dinamis adalah sebagai berikut:

- Karakteristikkan struktur solusi optimal.
- Definisikan secara rekursif nilai solusi optimal.
- Hitung nilai solusi optimal secara maju atau mundur.
- Konstruksi solusi optimal.

Relasi rekurens keuntungan optimal:

$\boldsymbol{f_1(x_1) = max\{R_{1,m1}\}}$ **untuk i = 1**

$\boldsymbol{f_i(x_i) = max\{R_{i,mi} + f_{i-1}(x_i - c_{i,mi})\}}$ **i = 2, 3,…,n**

m_i = nomor alternative untuk tiap tahap (i)

i = label untuk tahap sekarang (i = 1, 2, … , n)

$f_i(x_i)$ = kontribusi tahap i,i+1,…,n

$R_{i,mi}$ = hasil/pendapatan (*yield revenues*) dari tahap ke- i

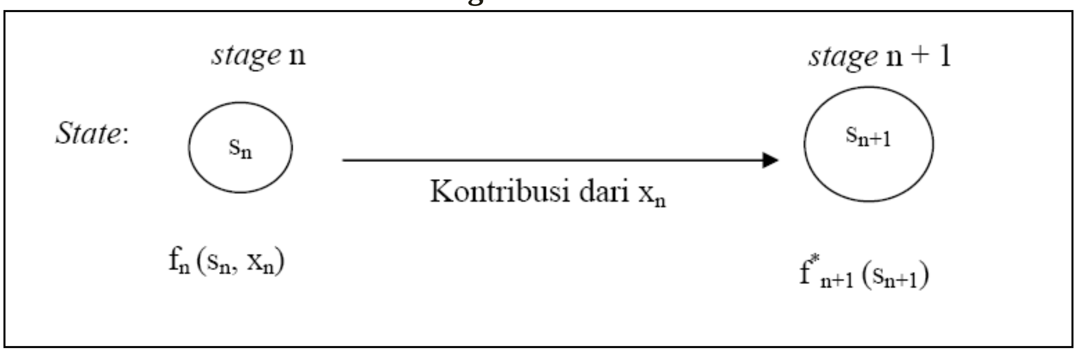
X_i = alokasi terbesar dari tahap ke- i

$C_{i,mi}$ = biaya (*cost*) yang akan dialokasikan

3. Pemrograman Dinamis Deterministik

Pemrograman deterministik dapat diuraikan dengan diagram yang ditunjukkan pada gambar di bawah ini:

Gambar 1. Struktur Dasar Pemrograman Dinamis Deterministik



Sumber : Hiller, Frederick, S, 1994

Keterangan :

Stage n menunjukkan tahapan ke n

State n menunjukkan status n

s_n menunjukkan keadaan sekarang untuk tahap n

s_{n+1} menunjukkan keadaan sekarang untuk tahap n+1

x_n menunjukkan peubah keputusan untuk tahap ke n

$f_n(s_n, x_n)$ menunjukkan kontribusi tahap n

$f^*_{n+1}(s_{n+1})$ menunjukkan kontribusi optimal

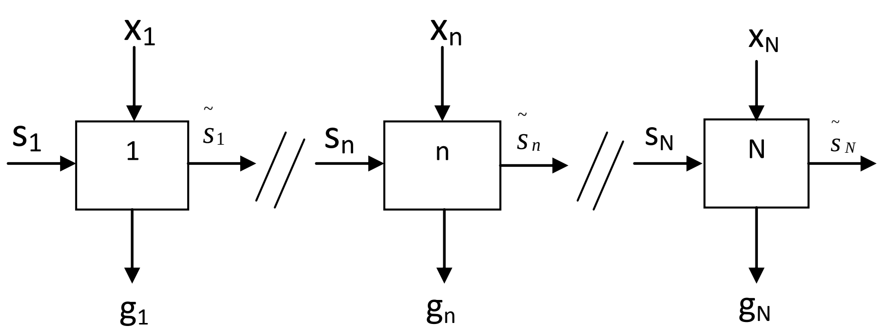
Dari Gambar 1. di atas dapat diuraikan sebagai berikut : Pada tahap n proses akan berada pada suatu keadaan s_n. Pembuatan keputusan kebijakan x_n selanjutnya menggerakkan proses ke keadaan s_{n+1} pada tahap (n+1). Kontribusi sesudahnya terhadap fungsi tujuan di bawah kebijakan optimal telah dihitung sebelumnya sebagai $f^*_{n+1}(s_{n+1})$. Keputusan kebijakan x_n juga memberi beberapa kontribusi kepada fungsi tujuan. Kombinasi kedua nilai ini dengan benar akan memberikan $f_n(s_n, x_n)$, yaitu kontribusi n tahap ke depan kepada fungsi tujuan. Pengoptimalan terhadap $x_n f_n^*(s_n) = f_n(s_n,x_n^*)$. Setelah ditemukan x_n^* dan $f_n^*(s_n)$ untuk setiap nilai s_n, prosedur penyelesaian sekarang siap bergerak mundur satu tahap Hiller, Frederick, S, 1994).

Dalam program deterministik dikenal juga rekursif maju dan rekursif mundur yang mana hubungan rekursif akan selalu memiliki bentuk

$$f_n^*(s_n) = \max\{f_n(s_n, x_n)\} \;\; atau \;\; f_n^*(s_n) = \min\{f_n(s_n, x_n)\}$$

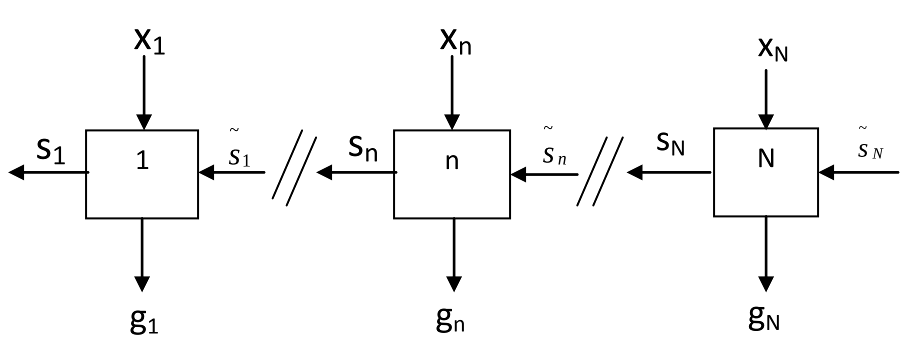
a. Rekursif maju

Proses perhitungan mulai dari tahap $n = 1$ hingga tahap n = N



b. Rekursif mundur

Proses perhitungan mulai dari tahap $n = N$ hingga tahap $n = 1$



c. Fungsi Transisi

- Rekursi Maju

$$s_{n-1} = s_n \otimes x_n$$

- Rekursi Mundur

$$s_{n+1} = s_n \otimes x_n$$

Tabel 1. Fungsi Transisi :

$\otimes$?	*Fungsi Transisi*	
	Maju	*Mundur*
+	$S_{n-1} = S_n + x_n$	$S_{n+1} = S_n + x_n$
-	$S_{n-1} = S_n - x_n$	$S_{n+1} = S_n - x_n$
$\times$	$S_{n-1} = S_n x_n$	$S_{n+1} = S_n x_n$
$\pm\sqrt{}$	$S_{n-1} = S_n \pm \sqrt{x_n}$	$S_{n+1} = S_n \pm \sqrt{x_n}$

d. Hubungan Rekursif maju :

$$f_n(S_n, x_n) = r_n(S_n, x_n) \otimes f_{n-1}^*(S_{n-1}), \qquad n = 1, \dots, N$$

$$f_n^*(S_n) = f(S_n, x_n^*), \qquad n = 1, \dots, N$$

$$f_n^*(S_n) = opt\left\{r_n(S_n, x_n) \otimes f_{n-1}^*(S_{n-1})\right\} \qquad n = 1, \dots, N$$

$$f_n^*(S_0) = 0$$

$$S_{n-1} = S_n \otimes x_n$$

e. Hubungan rekursif mundur

$$f_n(S_n, x_n) = r_n(S_n, x_n) \otimes f_{n+1}^*(S_{n+1}), \qquad n = N, \dots, 1$$

$$f_n^*(S_n) = f(S_n, x_n^*), \qquad n = N, \dots, 1$$

$$f_n^*(S_n) = opt\left\{r_n(S_n, x_n) \otimes f_{n+1}^*(S_{n+1})\right\} \qquad n = N, \dots, 1$$

$$f_{N+1}^*(S_{N+1}) = 0$$

$$S_{n+1} = S_n \otimes x_n, \qquad n = N, \dots, 1$$

Sesuai dengan penjelasan sebelumnya di bagian pendahuluan, terdapat beberapa contoh masalah pemrograman dinamis deterministik yaitu :

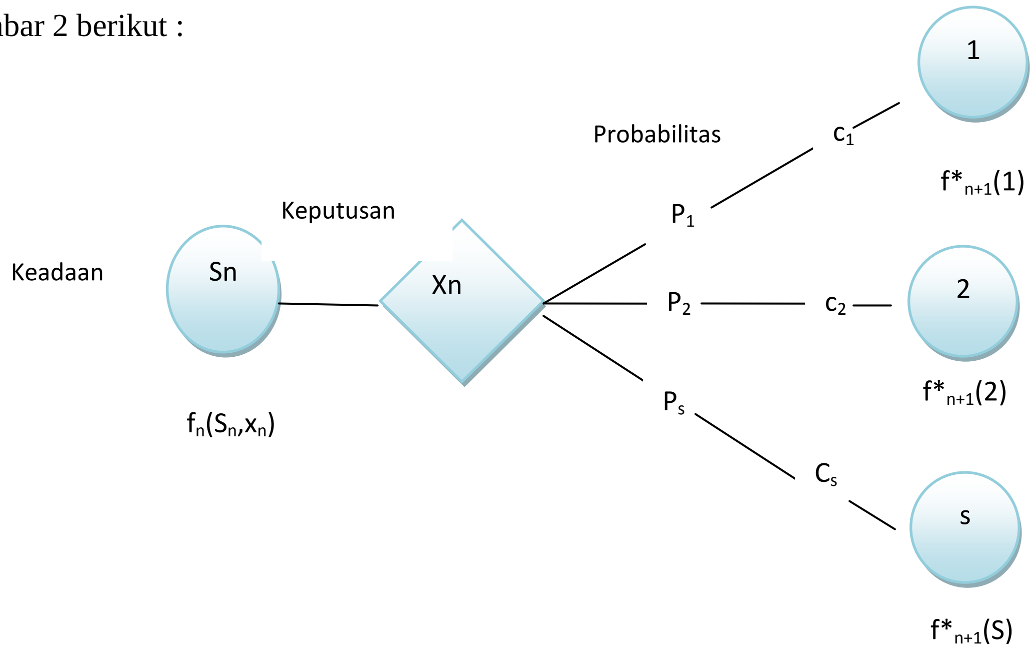
- ❖ Untuk status diskrit tunggal terdiri atas
  - *Stagecoach problem*
  - *Cargo loading problem*
  - *Inventory problem*
  - *Reliability problem*
- ❖ Untuk status kontinu tunggal contohnya adalah *Nonlinear programming problem*
- ❖ Untuk status kontinu majemuk contohnya *Linear Programming Problem*

4. Dinamik Program Probabilistik

Pemrograman dinamis probabilistik berbeda dengan pemrograman dinamis deterministic. Di mana pemrograman dinamis deterministik, pada tahap berikutnya sepenuhnya ditentukan oleh keadaan dan keputusan kebijakan pada tahap sebelumnya. Sedangkan pemrograman dinamis probabilistik, terdapat suatu distribusi probabilitas tentang keadaan mendatang. Akan tetapi, distribusi peluang ini tetap ditentukan oleh keadaan dan keputusan kebijakan pada keadaan sebelumnya. Jadi, terdapat dua hal dalam pemrograman dinamis probabilistik, yaitu :

- ❖ stage berikutnya tidak seluruhnya ditentukan oleh state dan keputusan pada stage saat ini, tetapi ada suatu distribusi kemungkinan mengenai apa yang akan terjadi.
- ❖ Walau demikian, distribusi kemungkinan ini masih seluruhnya ditentukan oleh state dan keputusan pada stage saat ini.

Struktur dasar yang dihasilkan pemrograman dinamis probabilistic diuraikan secara diagram dalam Gambar 2 berikut :



Gambar 2. Struktur pemrograman dinamis probabilistik

Di mana :

- S : melambangkan banyaknya keadaan yang mungkin pada tahap (*stage*) n + 1 dan keadaan ini digambarkan pada sisi sebelah kanan sebagai 1,2,…,S.
- (p_1,p_2,...p_s) adalah distribusi kemungkinan dari terjadinya suatu state berdasarkan state sn dan keputusan xn pada stage n
- ci adalah kontribusi dari stage n terhadap fungsi tujuan, jika state berubah menjadi state i
- *fn*(*Sn*, *xn*) menunjukkan jumlah ekspektasi minimum dari tahap *n* ke depan, dengan diberikan status dan keputusan pada tahap *n* masing-masing *Sn* dan *xn*.

Karena adanya struktur probabilistic, hubungan antara fn(s_n,x_n) dan f^*_{n+1}(s_n,x_n) agak lebih rumit dari pada untuk pemrograman dinamis deterministic. Bentuk yang tepat dari hubungan tergantung pada bentuk fungsi tujuan secara umum.

Dalam pemrograman dinamis probabilistic juga terdapat hubungan rekursif yang mengidentifikasi kebijakan optimal. Ada dua macam prosedur rekursif yaitu :

- *Forward recursive equation* (perhitungan dari depan ke belakang). Program dinamis bergerak mulai dari tahap 1 sampai tahap *n*. Peubah keputusan adalah x_1, x_2, …, x_n.

- *Backward recursive equation* (perhitungan dari belakang ke depan). Program dinamis bergerak mulai dari tahap *n*, terus mundur ke tahap $n - 1$, $n - 2$, dan seterusnya sampai tahap 1. Peubah keputusan adalah x_n, x_{n-1}, …, x_1.

Untuk ilustrasi, misalkan tujuannya adalah meminimumkan jumlah yang diharapkan dari konstribusi tahap-tahap secara terpisah. Pada kasus ini $fn(s_n,x_n)$ menggambarkan jumlah minimum yang diharapkan dari tahap *n* dan seterusnya, bila diketahui bahwa keadaan dan keputusan kebijakan pada tahap n adalah s*n* dan x*n* . Akibatnya,

$$fn(s_n,x_n) = \sum_{i=1}^{n} p_i[C_i + f_{n+1}(i)],$$

dengan $$f^*_{n+1}(i) = minimum\ f_{n+1} = minimum\ \{f_{n+1}(i, x_{n+1})\}$$

di mana peminimuman ini dibuat di atas nilai kelayakan bagi x_{n+1}

Salah satu contoh Permasalahan dalam Pemrograman Dinamis Probabilistic adalah :

- *Game (*Permainan)
- *Penambahan Penolakan (Reject Allowence)*

Perusahaan menerima order untuk membuat satu *item* dari suatu jenis produk tertentu.Karena pemesan menetapkan standar kualitas yang ketat, perusahaan harus memproduksi lebih dari satu *item* agar produk dapat diterima.

Jumlah tambahan *item* yang diproduksi disebut *reject allowance*.

5. Contoh Soal Problem Dinamik

a. Pertimbangkan masalah penganggaran modal, masing-masing pabrik memilih suatu langkah di mana suatu keputusan dibuat. Alternatif diberikan oleh variabel keputusan m_i pada langkah ke-i yang dicalonkan sebagai suatu rencana perluasan spesifik. Dalam hal ini fungsi pengembalian adalah $R_{i,mi}$.

Apa yang menggambarkan keadaan pada langkah ke-i? Catat bahwa langkah "Link" dengan fakta bahwa semua pabrik-pabrik (langkah-langkah) sedang bersaing untuk suatu bagian menyangkut modal yang terbatas C. Situasi ini menunjukkan bahwa keadaan harus digambarkan dalam kaitannya dengan alokasi modal.

Pengalaman telah menunjukkan bahwa seorang pemula dalam program dinamis biasanya akan menggambarkan keadaan ke-i sebagai "besarnya modal yang dialokasikan ke langkah i". Untuk melihat mengapa definisi ini tidak benar, pertimbangkan cara di mana masalah akan dipecahkan.

Definisi dari keadaan perlu mengizinkan seseorang untuk membuat suatu kemungkinan keputusan untuk keadaan sekarang tanpa mengecek keputusan yang dibuat untuk langkah-langkah sebelumnya. Definisi dari keadaan di atas hanya menunjukkan bahwa jumlah yang dialokasikan ke langkah i dapat sekecil nol atau setinggi total modal tersedia C. Ini bukanlah informasi yang cukup untuk menjamin sesuatu yang mungkin untuk langkah yang sekarang. Sebagai contoh, misalnya diputuskan untuk mengalokasikan 0.4C dolar kepada langkah yang sekarang. Kelayakan dari keputusan ini tidaklah dijamin tanpa mengecek langkah-langkah yang terdahulu untuk memastikan bahwa alokasi modal total mereka tidak melebihi C-.0,4C= 0,6 C dolar. Hal ini menunjukkan bahwa langkah yang sekarang tidaklah optimal dengan bebas, bertentangan dengan gagasan dasar dari program dinamis.

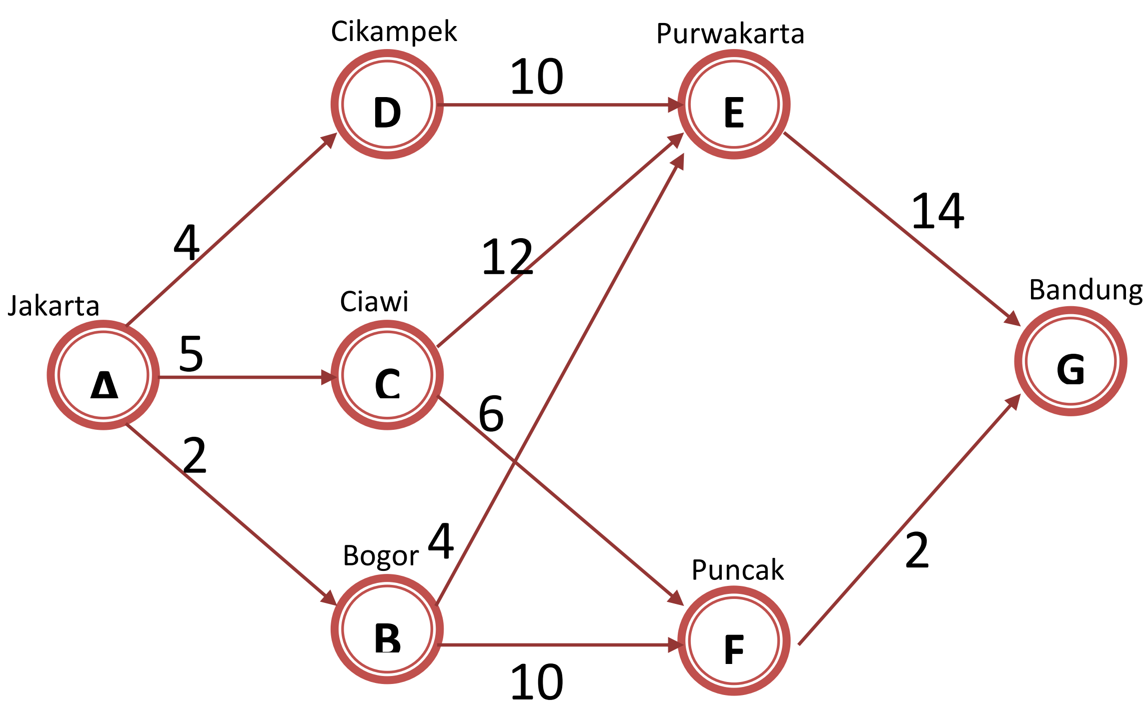
Pertimbangkan definisi dari keadaan pada langkah yang I katakan sebagai jumlah modal yang dialokasikan ke langkah-langkah 1,2,…,dan I (itu adalah, langkah pertama I)". Definisi ini benar, sejak perbedaan antara alokasi modal kepada langkah pertama I, keadaan dari sistem pada langkah i dan alokasi modal ke langkah (i-1), keadaan dari sistem pada langkah (i-1) memberikan jumlah modal untuk dialokasikan ke langkah yang I saja. Ini

contoh menunjukkan, mengizinkan seseorang untuk membuat suatu kemungkinan keputusan untuk langkah I tanpa mengecek terlebih dahulu langkah (i-1).

Masalah penganggaran modal menghadirkan suatu masalah alokasi yang khas di mana suatu sumber daya (atau, umumnya, sumber daya) didistribusikan (secara optimal) antarsejumlah aktivitas (langkah). Definisi dari keadaan menyangkut sistem untuk semua permasalahan alokasi biasanya sama, yakni jumlah dari sumber daya yang dialokasikan ke suatu jumlah berurutan dari langkah-langkah yang mulai dari langkah 1. Jenis lain dari permasalahan tidak jatuh masuk ke kategori ini, bagaimanapun, dan memerlukan suatu definisi yang berbeda menyangkut keadaan dari system

b. Contoh Permasalahan Rute Terpendek

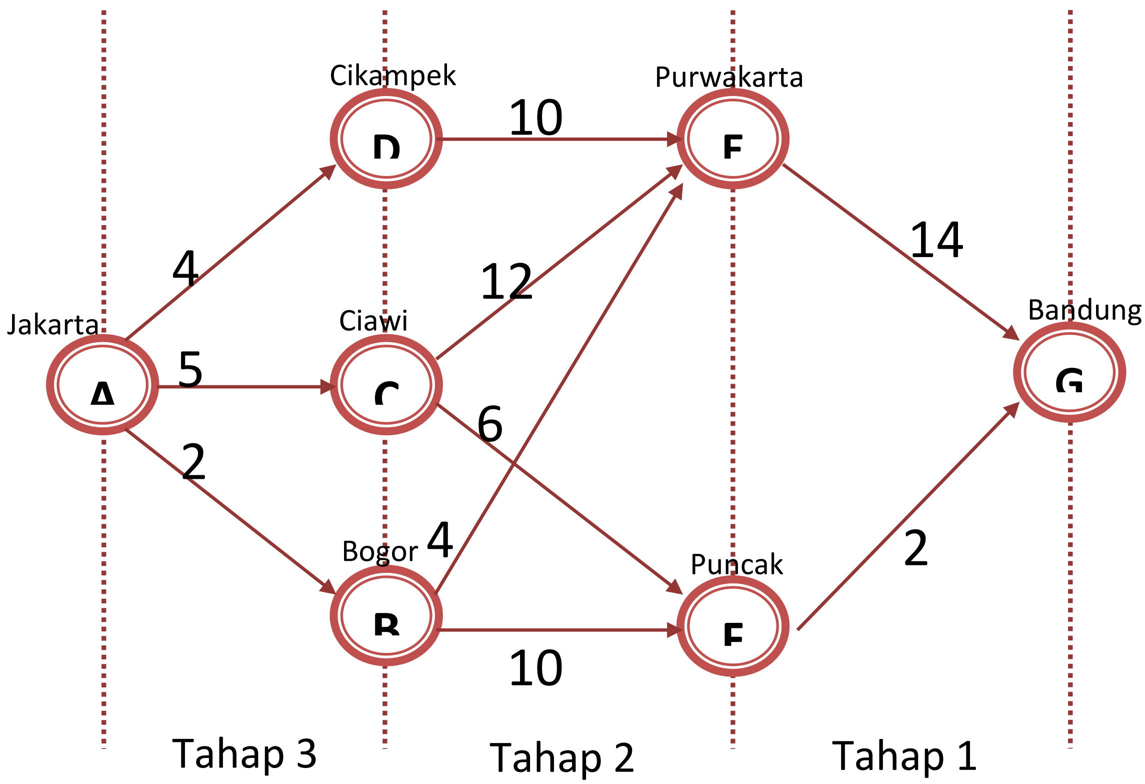
Agus akan melakukan perjalanan dari Jakarta menuju Bandung, dengan melewati kota/tempat seperti terlihat di peta perjalanan Gambar 3. Tanda bulatan (node) mewakili kota, tanda panah (busur) mewakili jalan raya antar kota/tempat, jarak antar kota tertulis di tanda panah.



Gambar 3. Peta Perjalanan

Langkah 1 :

Tahap pertama adalah membagi permasalahan menjadi subproblem. Gambar 2 menunjukkan tahapan dalam problem ini. Dalam pemrograman dinamis, biasanya dimulai dari bagian terakhir problem, sebagai tahap 1, dan bekerja mundur sampai dengan permulaan problem atau jaringan. Tabel 1 menunjukkan jarak busur antar tahapan.



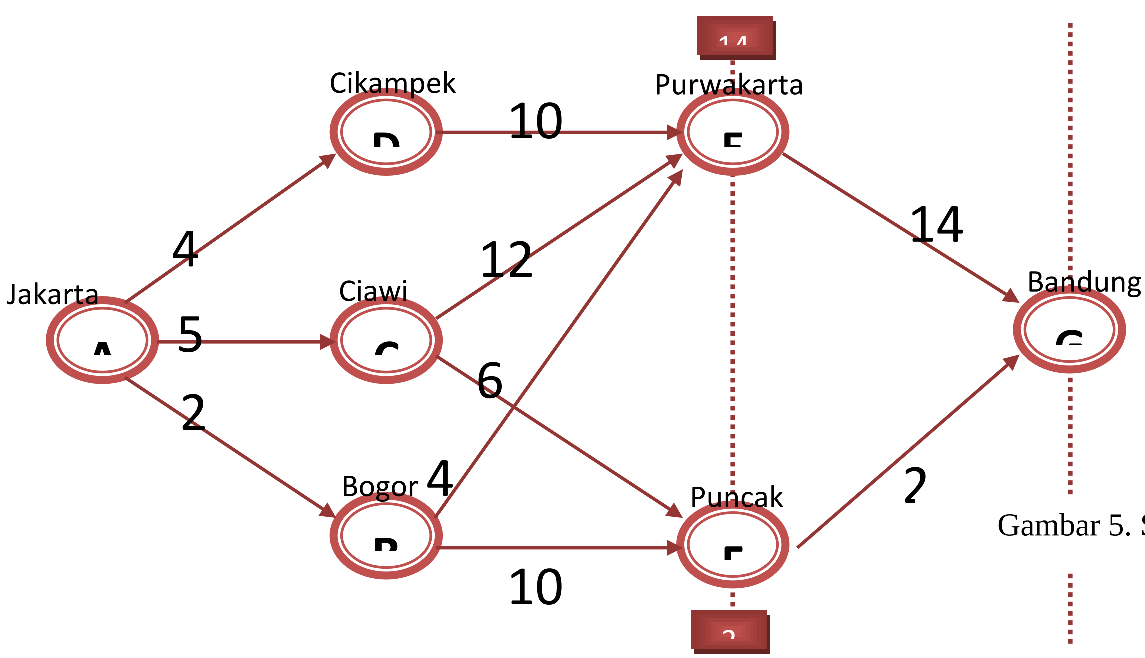
Gambar 4. Tahapan Pencarian Rute Terpendek

Tabel 4. Jarak Tiap Busur

Tahap	Busur	Jarak
1	E – G	14
	F – G	2
2	D – E	10
	C – E	12
	C – F	6
	B – E	4
	B – F	10
3	A – D	4
	A – C	5
	A – B	2

Langkah 2 :

Pada langkah berikut, kita memecahkan tahap 1, bagian terakhir dari jaringan. Biasanya bagian ini mudah dilakukan. Kita cari jalur terpendek ke akhir jaringan, dalam problem ini node G. Pada tahap 1, jalur terpendek hanya terdiri dari node E ke node G dan node F ke node G. Kita gambarkan jarak minimum tersebut dalam kotak di node awal di tahap 1, yaitu node E dan node F.



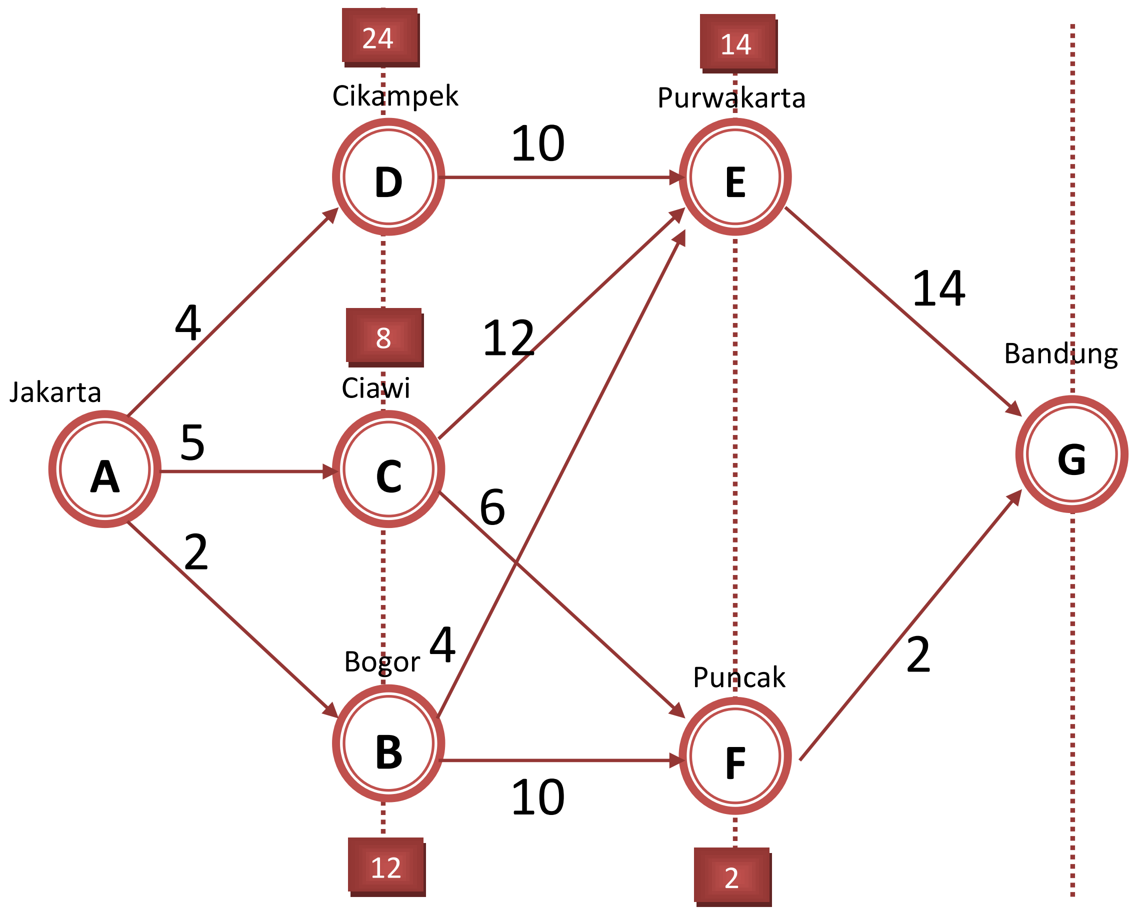
Gambar 5. Solusi Tahap 1

Tahap 1		
Node awal	**Jarak Terpendek ke Node G**	**Busur dalam jalur**
E	14	E-G
F	2	F-G

Langkah 3 :

Solusi tahap 2 dapat dilihat di Gambar 6.

Pada node D, jarak terpendek ke node G adalah busur D-E dan E-G dengan jarak total 24. Pada node C, jalur terpendek adalah busur C-F dan F-G dengan jarak minimum 8. Pada node B, jalur terpendek adalah B-F dan F-G dengan minimum 12.

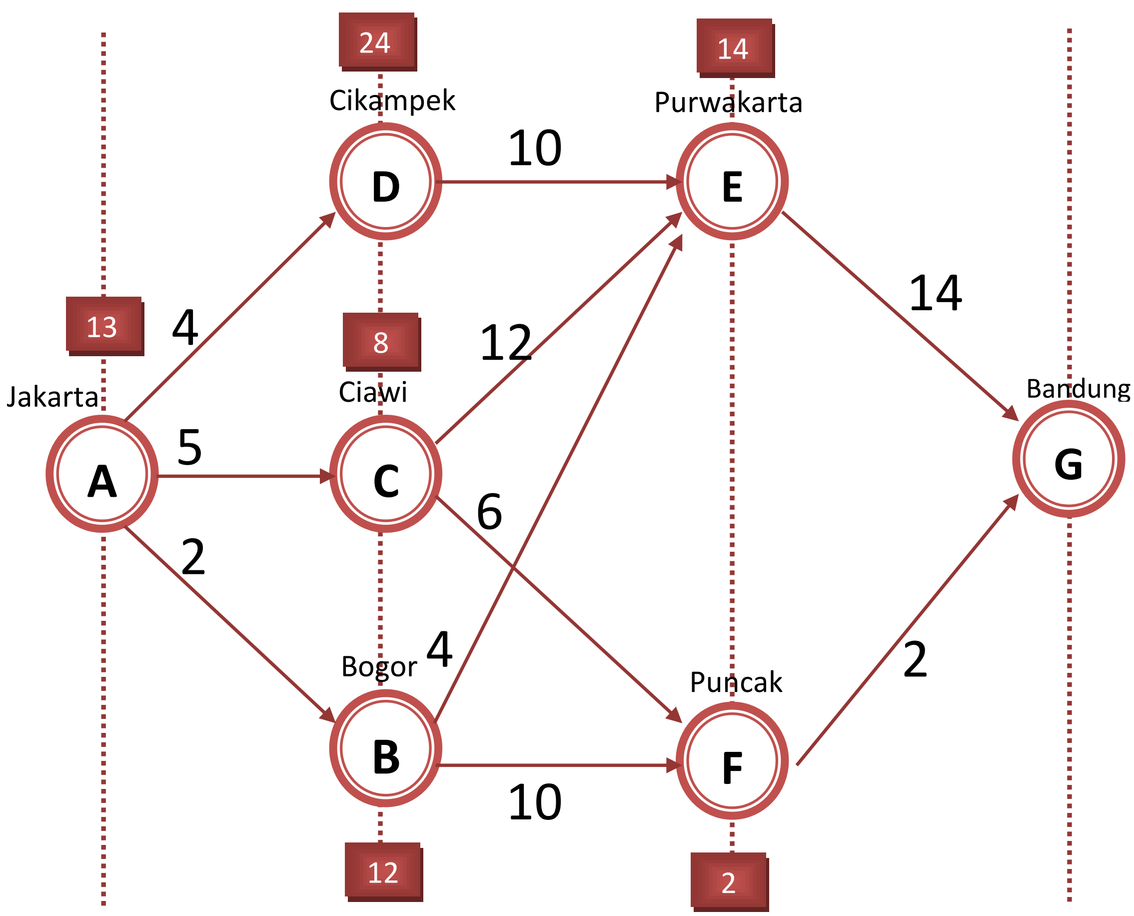


Gambar 6. Solusi Tahap 2

Tahap 2		
Node awal	**Jarak terpendek ke node G**	**Busur dalam jalur**
D	24	D-E E-G
C	8	C-F F-G
B	12	B-F F-G

Untuk mendapatkan solusi optimal pada tiap-tiap tahap, yang kita harus perhitungkan adalah busur ke tahap berikutnya dan solusi optimal tahap berikutnya. Pada tahap 3, kita hanya perlu memperhitungkan 3 busur yang mengarah ke tahap 2 (A-B, A-C, dan A-D) dan hasil optimal di tahap 2, yang telah dicatat dalam table sebelumnya.

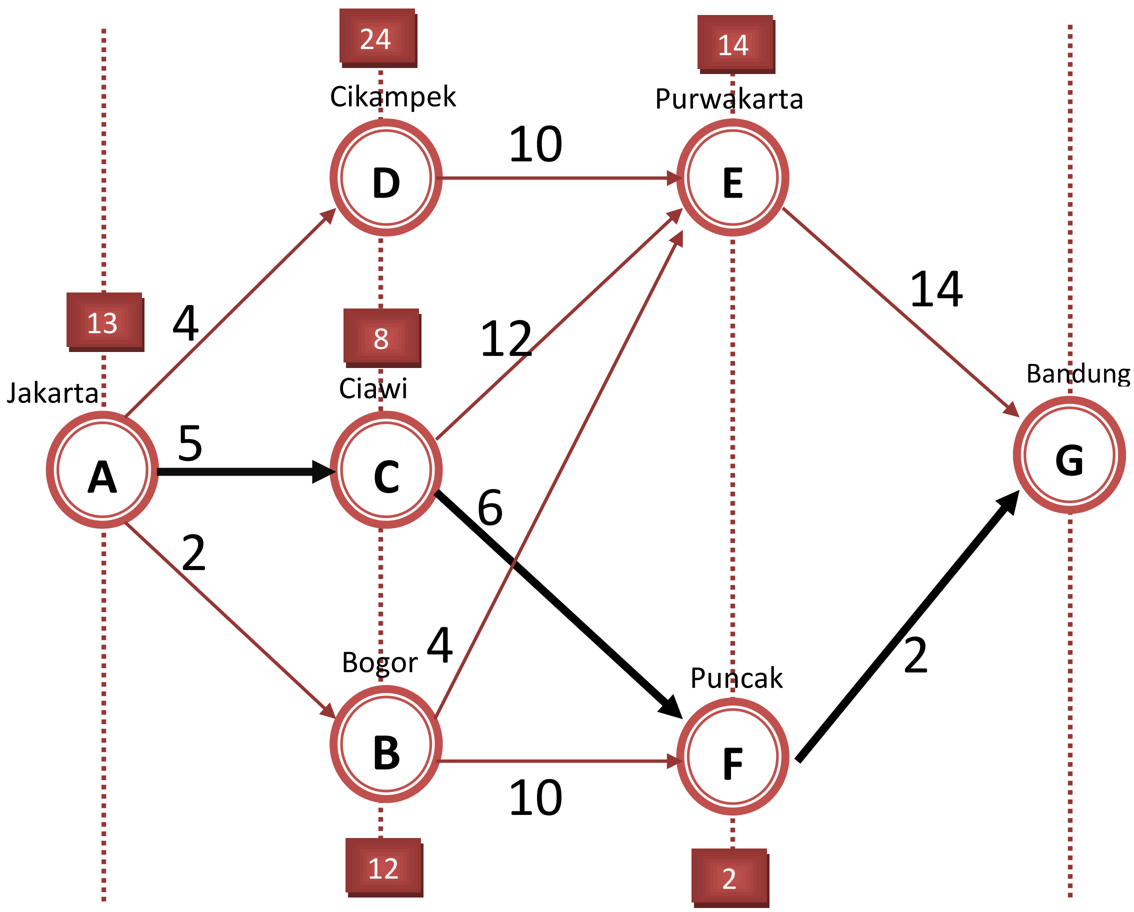
Solusi tahap 3 terlihat pada Gambar 7.



Gambar 7. Solusi Tahap 3

Tahap 3		
Node Awal	**Jarak terpendek ke node G**	**Busur dalam Jalur**
A	13	A – C C – F F – G

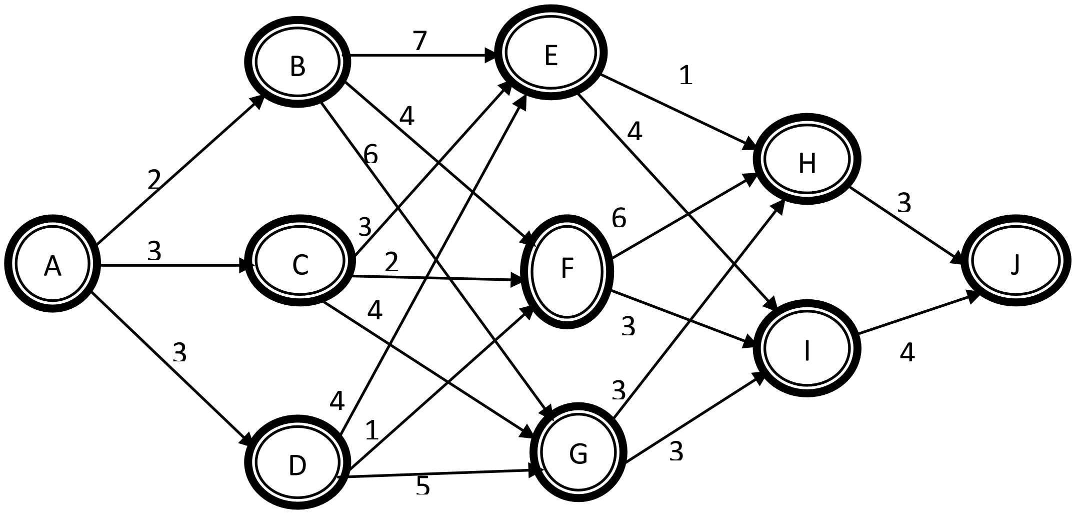
Solusi akhir dari problem tersebut adalah seperti terlihat di table di atas, jalur yang ditempuh adalah A – C, C – F, dan F – G seperti terlihat di Gambar 8, dengan panah dicetak tebal.



Gambar 8. Solusi Akhir

Jadi, rute/jarak terpendek yang dapat ditempuh oleh Agus adalah **Jakarta – Ciawi – Puncak-Bandung.**

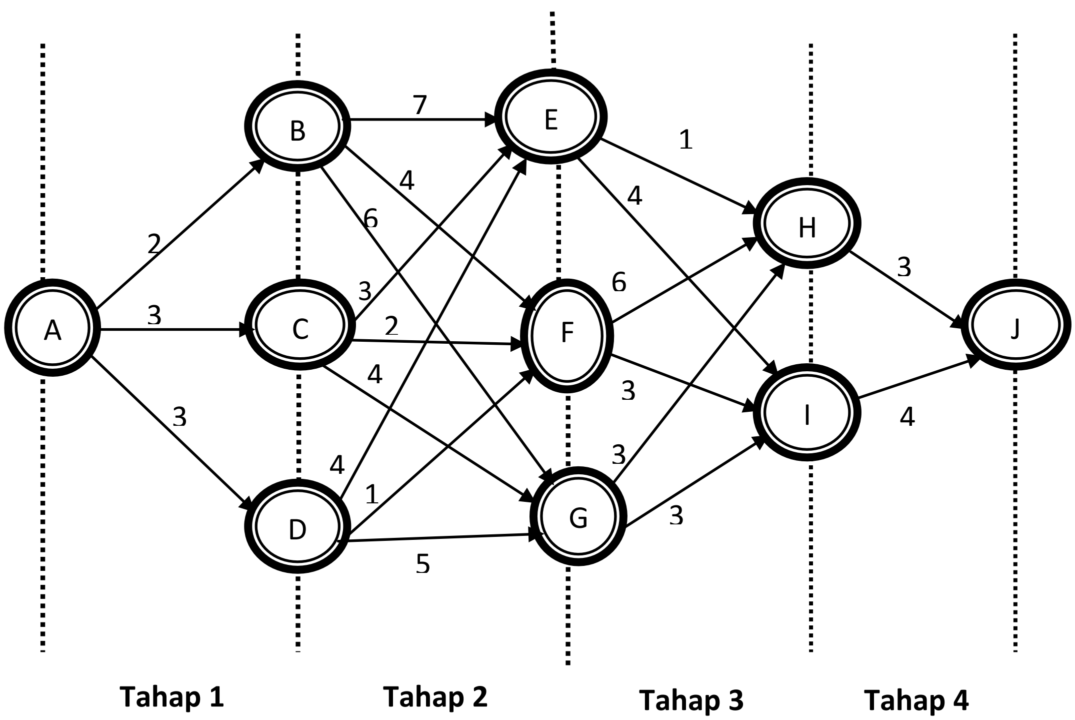
c. Masalah **Stagecoach Problem**



Tentukan lintasan yang memberikan jarak paling minimum !

Jawab :

Gambar di atas dapat dibagi menjadi 4 tahap



Status :

S_n = simpul saat ini pada tahap *n*

Variabel keputusan

X_n = simpul tujuan pada tahap *n*

Fungsi pengembalian

$g_n = c_{(s_{n,x_n})}$

$c_{(s_{n,x_n})}$ = jarak dari simpul S_n ke simpul X_n

Fungsi transisi (mundur):

$S_{n+1} = X_n$

Hubungan rekursif (mundur):

$\boldsymbol{F_n^*(S_n) = \min \left\{ c_{(s_{n,x_n})} + f_{n+1}^* \quad (s_{n+1}) \right\},\ n = 4,\ldots 1}$

$F_n^*(S_n) = \min \left\{ c_{(s_{n,x_n})} + f_{n+1}^* \quad (x_n) \right\},\ n = 4, \ldots 1$

$F_5^*(S_5) = 0$

Maka untuk melakukan perhitungannya dapat dilakukan dengan rekursif mundur yang berarti dimulai dari titik tujuan sampai ke titik start

- ***Tahap n =4***

$F_5^*(S_5) = 0$ (Bernilai nol karena tujuan akhir rute perjalanan di atas hanya sampai tahap ke 4)

Pada tahap ini, yang pertama kali ditinjau adalah simpul tujuannya dalam hal ini simpul J karena perhitungannya menggunakan rekursif mundur. Selanjutnya dari simpul J terdapat 2 percabangan rute yaitu dari H-J dan I-J. Adapun jaraknya ($c_{(s_{n,x_n})}$ adalah

Untuk H−J : 3

Untuk I−J : 4

Sehingga nilai f_4^* (S_4) untuk H−J = 3 dan I− J= 4 sedangkan x_4^* untuk simpul H dan I adalah J. Adapun hasilnya dapat dilihat pada **Tabel 1** di bawah ini :

Tabel 1: Nilai Optimum dalam Tahapan ke 4

S_4 \ x_4	$f_4(S_4, x_4) = c_{(S_4,x_4)}$	$f_4^*(S_4)$	x_4^*
	J		
H	*3*	*3*	*J*
I	*4*	*4*	*J*

- ***Tahap n = 3***

Di tahap ini, yang menjadi x_3 adalah H dan I sedangkan S_3 adalah E, F, dan G. Selanjutnya kita meninjau jumlah jarak perjalanan dari E−H, F−H, G−H, I−E, I−F, I−G dengan menggunakan rumus $f_3(S_3, x_3) = c_{(S_3, x_3)} + f_4^*(x_3)$ sehingga diperoleh

Untuk E−H = 4	Untuk E−I= 8
F−H= 9	F−I= 7
G−H= 6	G−I= 7

Dari semua jumlah jarak yang diperoleh, dicari nilai jarak yang paling minimum pada tahap ke 3 ($f_3^*(S_3)$) dengan membandingkan

1. Jarak antara E−H dan E−I, nilai yang paling minimum adalah 4 dengan $x_3^* = H$
2. Jarak antara F−H dan F−I, nilai yang paling minimum adalah 7 dengan $x_3^* = I$
3. Jarak antara G−H dan G−I, nilai yang paling minimum adalah 6 dengan $x_3^* = H$

Adapun hasilnya, dapat dilihat dalam **Tabel 2** seperti di bawah ini :

Tabel 2: Nilai Optimum dalam Tahapan ke 3

S_3 \ x_3	$f_3(S_3, x_3) = c_{(S_3,x_3)} + f_4^*(x_3)$		$f_3^*(S_3)$	x_3^*
	H	*I*		
E	*1+3=4*	*4+4=8*	*4*	***H***
F	*6+3=9*	*3+4=7*	*7*	*I*
G	*3+3=6*	*3+4=7*	*6*	*H*

- **Tahap n = 2**

Seperti halnya pada tahap sebelumnya, pada tahap n = 2 ini akan dcari nilai $f_2(S_2, x_2)$. Dalam hal ini akan dicari jumlah yang harus ditempuh yaitu pada simpul B−E, B−F, B−G, C−E, C−F, C−G, D−E, D−F, dan D−G dengan melibatkan jumlah jarak minimum yang telah ditentukan sebelumnya (pada tahap 4 dan 3) atau dapat dituliskan dalam rumus :

$$f_2(S_2, x_2) = c_{(S_2,x_2)} + f_3^*(x_2)$$

Sehingga diperoleh nilai-nilainya

Untuk S_2=B maka B−E = 11
B−F = 11
B−G = 12

Nilai yang paling minimum adalah 11 dengan x_2^* = E,F

UntukUntuk S_2= C maka C−E = 7
C−F = 9
C−G = 10

Nilai yang paling minimum adalah 7 dengan x_2^* = E

Untuk S_2= D maka D−E = 8
D−F = 8
D−G = 11

Nilai yang paling minimum adalah 8 dengan x_2^* = E,F

Adapun hasilnya dapat dilihat pada **Tabel 3** seperti gambar di bawah ini :

Tabel 3: Nilai Optimum dalam Tahapan ke 2

x_2 / S_2	$f_2(S_2, x_2) = c_{(S_2,x_2)} + f_3^*(x_2)$			$f_2^*(S_2)$	x_2^*
	E	*F*	*G*		
B	*7+4 = 11*	*4+7=11*	*6+6=12*	*11*	*E,F*
C	*3+4 = 7*	*2+7=9*	*4+6=10*	*7*	*E*
D	*4+4 = 8*	*1+7=8*	*5+6=11*	*8*	*E,F*

- **Tahap n = 1**

Pada tahap ini yang juga merupakan tahap terakhir dari rekursi mundur, akan ditunjukkan jarak yang dapat ditempuh dari simpul A sebagai S_1 menuju x_1 dalam hal ini simpul B, C, D. Hal ini dapat dilakukan dengan menjumlahkan jarak dari S_1 ke x_1 dengan nilai minimum dari f_2 atau dapat dirumuskan sebagai berikut :

$$f_1(S_1, x_1) = c_{(S_1, x_1)} + f_2^*(x_1)$$

Dan diperoleh

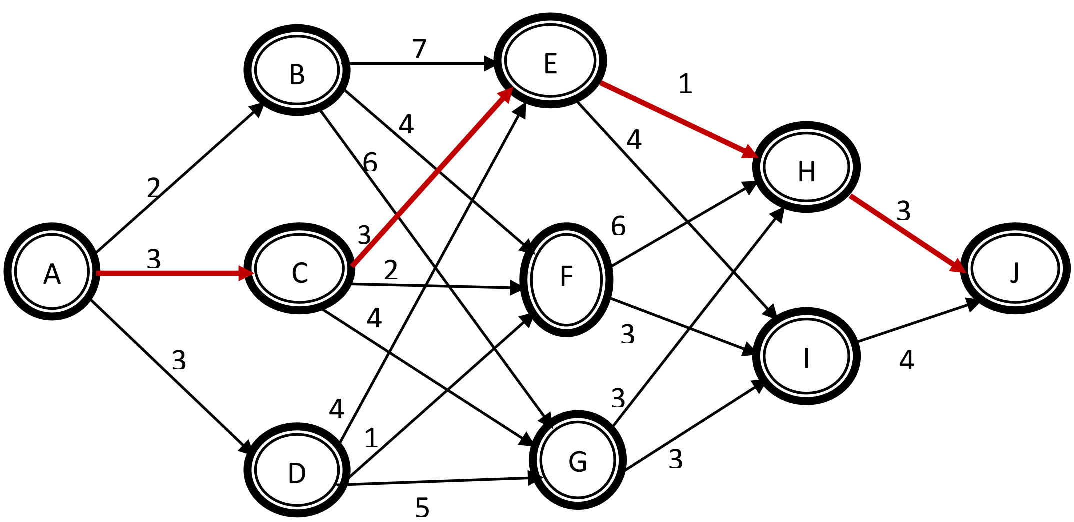
A−B = 13, A−C = 10, A−D = 11

Dari ketiga nilai yang telah diperoleh, maka nilai yang paling minimum $(f_1^*(S_1) = 10\ dengan\ x_1^*$ = C. Adapun hasilnya dapat ditunjukkan dalam tabel seperti pada **Tabel 4** sebagai berikut :

Tabel 4

S_1 \ x_1	$f_1(S_1, x_1) = c_{(S_1, x_1)} + f_2^*(x_1)$			$f_1^*(S_1)$	x_1^*
	B	*C*	*D*		
A	*2+11 = 13*	*3+7 = 10*	*3+8 = 11*	*10*	*C*

Jadi dari seluruh tahap yang telah dilakukan maka diperoleh rute perjalanan yang memiliki jarak paling minimum yaitu dari A→C→E→H→J dengan jumlah jaraknya adalah 10



6. Soal Mandiri dan Tim

a. Soal: halaman 432 no : 22 frederick S Hillier, 1994

22. seorang pemain backgamon akan bermain 3 pertandingan secara berurutan dengan temen-temannya malam ini. Untuk setiap pertandingan dia akan memiliki kesempatan untuk memasang taruhan seimbang bahwa ia akan menang; banyaknya taruhan dapat berupa jumlah sembarang menurut pilihannya antara 0 dan jumlah uang yang masih dipunyainya setelah bertaruh pada pertandingan sebelumnya. Untuk setiap pertandingan peluangnya untuk menang adalah ½ sehingga ia akan memenangkan jumlah taruhan, sedangkan peluang kalahnya adalah ½ sehingga ia akan menghilangkan jumlah taruhannya. Ia akan memulai taruhannya dengan $75 dan tujuannya adalah untuk mencapai $100 pada akhir permainan. (karena permainan ini antar teman, dia tidak ingin mengakhiri pertandingan dengan uang lebih dari $100). Dengan demikian, dia ingin mendapatkan kebijakan taruhan optimal (termasuk seri) yang memaksimumkan peluangnya untuk mendapatkan tepat $100 setelah 3 kali pertandingan.

Gunakan pemrograman dinamis untuk menyelesaikan masalah ini.

b. . Menghadapi pemilihan umum yang akan datang, seorang pemimpin partai menugaskan juru kampanye untuk berkampanye di 4 daerah pemilihan. Dia berpendapat, tidak efisien bila menugaskan seorang juru kampanye ke lebih dari satu daerah pemilihan. Tetapi dia tidak ingin menugaskan seorang jurukampanye ke suatu daerah pemilihan tertentu bila jurukampanye tersebut dapat melakukan lebih baik di tempat pemilihan lain. Tabel 1. berikut memperlihatkan perkiraan banyaknya penambahan pemilih bagi calon partai tersebut pada tiap daerah pemilihan jika dialokasikan berbagai jumlah juru kampanye. Berapa banyak juru kampanye harus ditugaskan ke setiap daerah pemilihan supaya dicapai total penambahan pemilih bagi calon partai tersebut maksimum?

Tabel 1, Data untuk juru kampanye di setiap daerah

Jumlah juru kampanye	DAERAH			
	1	2	3	4
0	0	0	0	0
1	25	20	33	13
2	42	38	43	24
3	55	54	47	32
4	63	65	50	39
5	69	73	52	45
6	74	80	53	50

c. Carilah dua soal untuk problem deterministic dan probabilistic untuk tugas Tim dan Mandiri di tekx book

DAFTAR PUSTAKA

Frederick, S. H, Gerald, 1990***." Introduction to Operation Research"***. Fifth edition, Mc Graw-Hill, New York.

Gass, Saul L, 1984**. "*Linear Programming : Method and Application"***. Fifth edition, Mc Graw-Hill, New York

Robert V. Hogg, Allent t. Craig, 1995. **" Introduction To Mathematical Statistics".** Fifth Edition, Prentice Hall, Inc Englewood

Ronal. E. Miller, 2000. "***Optimization" Foundation and Application.*** Prentice Hall, Inc USA

Taha, Hamdy, 1993. "***Operation Research"***. Fifth edition, Mc Graw- Hill, New York

FORMAT RENCANA PEMBELAJARAN

Kompetensi Utama : **Kemampuan dalam menguasai metode matematika, mengkomunikasi konsep-konsep, dan penerapan dalam bidang industri, ekonomi, pertanian serta lainnya (No 2, 3, 4, 5, / C1, C2, C3, C4).**

Kompetensi Pendukung :**Kemampuan membuat model matematika, penguasaan software saintifik, penerapan statistika, komputasi dalam bidang industri, ekonomi, pertanian dan lainnya, membuat laporan, presentasi, (No 5, 8, 9, 10, 11 / C1, C2, C3, C4).**

Kompetensi Lainnya : **Kemampuan berkomunikasi yang baik, mengembangkan diri dengan prinsip budaya bahari, beradaptasi dalam masyarakat dan bertanggung jawab (No 13, 14, 15, 16, / C1, C2, C3).**

Tabel 1. FORMAT RENCANA PEMBELAJARAN RISET OPERASI (Sem IV/ Wajib mhs Math)

(1) Minggu ke	(2) Entry Skill	(3) Materi Pembelajaran	(4) Strategi Pembelajaran	(5) Kemampuan akhir yang dicapai (Kompetensi)	(6) Kriteria penilaian (Indikator)	(7) Bobot nilai (%)
1	Mata kuliah Matematika dan Statistika	Penjelasan kontrak perkuliahan (tujuan, lingkup, materi, kegunaan pembelajaran serta kaitannya dengan mata kuliah lain) pada *Transportation Problem, Net Work Planning, Game Theory dan Dynamic Problem* Pembentukan kelompok (Tim)	kuliah	Memahami dengan baik terhadap tujuan, kegunaan dari materi kuliah, serta kaitannya dengan mata kuliah lain.	Tanya jawab	

Lanjutan

2 - 3		*Penjelasan defenisi, asumsi dari Problem Transportasi: (Model Tugas) *Penjelasan pendekatan sudut barat laut dan penyelesaian dasar awal. *Latihan soal	Kuliah Kajian Pustaka. Tugas. *(Cooperative learning*	Mamahami dengan baik tujuan, kegunaan dari model transportasi Mahasiswa mampu menyelesaikan masalah dengan sudut barat laut.	-Tanya jawab. -Mahasiswa mampu menyelesaikan tugas dengan baik. -Quis.	5
4 - 5		*Penjelasan Metode Vogel, Batu Loncatan dan Danzing. *Kontribusi dari Model Transfortasi. *Pengujian Optimalisasi dalam Transportasi *Latihan Mandiri, Tim	Kuliah. Kajian Pustaka. *Small group discussion.* Tugas mandiri dan Tim *(Cooperative learning).*	Mahasiswa memiliki wawasan dalam menganalisis masalah transportasi dengan metode Vogel, Batu Loncatan dan Danzing, serta mampu mengambil keputusan dari analisisnya.	Presentasi Tim, Tanya jawab (Debate). Mahasiswa mampu menentukan model yang terbaik dan akurat dan ketepatan pemakaiannya. Portofolio tiap Tim dan Mandiri	15
6 - 8		Penjelasan Analisis *Net Work Planning* : *Masalah lintasan terpendek *Menentukan Jalur kritis *Menghitung jalur Kritis *Masalah arus dengan biaya minimum. *Kontribusi dari grafik waktu dan tingkatan sumber. *Peluang dan ongkos penetapan dalam jadwal proyek. Perencanaan PERT-CPM	Kuliah Kajian Pustaka *Small group discussion.* Tugas Mandiri dan Tim *(Project base learnng).*	Mahasiswa memiliki wawasan dalam membuat proyek jaringan kerja dengan lintasan terpanjang dan terpendek. Mahasiswa mampu membuat diagram alir dari suatu proyek dengan lintasan terpanjang dan terpendek, serta mengambil keputusan yang akurat.	Mahasiswa mampu menganalisis dan menentukan lintasan yang terpendek, sesuai anggaran proyek. Presentasi Tim, Tanya jawab (Debate). Quis Portofolio tiap Tim dan Mandiri	20

Lanjutan

9 - 10		Penjelasan Game Theory : * Perumusan Permainan Dua orang jumlah nol, *Penyelesaian Permainan Sederhana dan Strategi Campuran. *Latihan soal Mandiri	Kuliah Kajian pustaka *Small group discustion* *(Cooperative learning)* Tugas mandiri	Mahasiswa memiliki wawasan dalam membuat permainan yang sederhana dan strategi campuran, serta mampu mengambil keputusan.	Mahasiswa mampu membuat jenin model permainan dan dapat menyelesaikann dengan pengambilan keputusan yang tepat. Tanya jawab (Debate)	10
11 - 12		Penjelasan Teori Permainan : *Prosedur Penyelesaian Grafik *Program linier dan Perluasan Permainan. *Soal latihan mandiri	Kuliah Kajian pustaka *Small group discustion* *(Cooperative learning)* Tugas mandiri	*Mahasiswa mempunyai wawasan dalam menjabarkan model permainan dengan grafik, program linier dan perluasannya.. *Mahasiswa dapat menyelesaikan tugas mandiri dengan baik.	*Mahasiswa mampu membuat model permainan dan menyelesaikan dengan cara gragik, program linier serta perluasan permainan *Tanya jawab, presentasi mandiri	15
13 - 14		Penjelasan Model Program Dinamik (PD): *Unsur dari model PD *Representasi bersama yang berulang/ contoh aplikasinya. *Program Deterministik (Perhitungan maju / mundur) * Program Probabilistik (Problem dari ukuran PD) *Soal latihan Mandiri dan Tim	Kuliah Kajian pustaka *Small group discustion* *(Project base learnng).* Tugas mandiri dan tim	Mahasiswa memiliki pengetahuan dalam membuat keputusan yang saling berhubungan (Dinamik) dari contoh portofolio, dan dapat membedakan antara program dinamik, deterministik serta probabilistik.	-Mahassiswa mampu membuat model masalah dinamis, deterministik,dan probabilitik dari kasus yang sederhana dengan benar dan sekaligus dapat menganalisis -Presentasi Tim -Tanya jawab -Quis, portofolio	15

Lanjutan

15 - 16		Penjelasan Program Bilangan Bulat (PBB) : *kemungkinan perumusan lain dengan peubah biner, *Penyelesaian dan perspektifnya. *Teknik percabangan dan pembatasan serta aplikasinya. * Algoritma percabangan dan pembatasan serta aplikasinya untuk PBB campuran. *Soal latihan Mandiri dan Tim	Kuliah Kajian pustaka *Small group discustion* *(Cooperative learning)* *(Project base learnng).*	Mahasiswa mempunyai pengetahuan membuat PBB dari kemungkinan perumusan lain dengan peubah biner, dapat membuat algoritma percabangan dan perbatasan untuk PBB campuran serta penyelesaian dan perspektifnya,	Mahasiswa mampu membuat model PBB dengan peubah biner dan menganalisisnya. Mahasiswa mampu membuat algoritma percabangan pada PBB campuran dan dapat menyelesaikannya pada kasus yang sederhana. Tanya jawab Presentasi Tim, Portofolio	**20**

Keterangan ; C1 = Remember ; C2 = Under standing ; C3 = Apply ; C4 = Analysis