# Praktikum 15

# Graph

# (Algoritma Singlepath)

## A. TUJUAN PEMBELAJARAN

Setelah melakukan praktikum dalam bab ini, mahasiswa diharapkan mampu:

1. Mengerti algoritma pencarian jalur terpendek untuk SinglePath
2. Mampu mengimplementasikan algoritma Djikstra

## B. DASAR TEORI

### B.1 Algoritma Djikstra

Algoritma Dijkstra, dinamai menurut penemunya, Edsger Dijkstra, adalah algoritma dengan prinsip greedy yang memecahkan masalah lintasan terpendek untuk sebuah graf berarah dengan bobot sisi yang tidak negatif.

Algoritma Dijkstra merupakan salah satu varian bentuk algoritma populer dalam pemecahan persoalan yang terkait dengan masalah optimasi. Sifatnya sederhana dan lempang (*straightforward*). Sesuai dengan arti greedy yang secara harafiah berarti tamak atau rakus - namun tidak dalam konteks negatif -, algoritma greedy ini hanya memikirkan solusi terbaik yang akan diambil pada setiap langkah tanpa memikirkan konsekuensi ke depan.

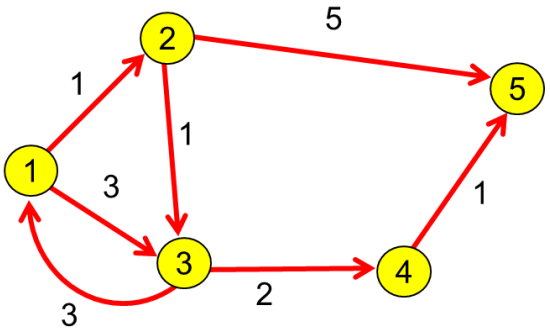
Input algoritma ini adalah sebuah *graph* berarah yang berbobot (*weighted directed graph*) G dan sebuah sumber vertex s dalam G dan V adalah himpunan semua vertices dalam *graph* G.

Setiap sisi dari *graph* ini adalah pasangan *vertices* (u,v) yang melambangkan hubungan dari vertex u ke vertex v. Himpunan semua tepi disebut E. Bobot (*weights*) dari semua sisi dihitung dengan fungsi : w: E → [0, ∞), jadi w(u,v) adalah jarak tak-

negatif dari vertex u ke vertex v. Ongkos (*cost*) dari sebuah sisi dapat dianggap sebagai jarak antara dua vertex, yaitu jumlah jarak semua sisi dalam jalur tersebut. Untuk sepasang vertex s dan t dalam V, algoritma ini menghitung jarak terpendek dari s ke t.

Langkah-langkah algoritma Djikstra adalah sebagai berikut :

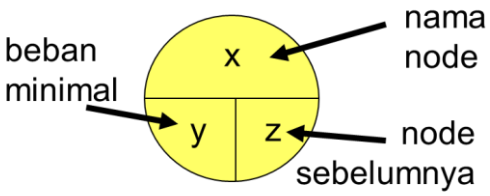
1. Tentukan titik asal dan titik tujuan sebelum proses
2. Akumulasikan jarak minimal dan simpan ke titik berikutnya.
3. Lakukan dari titik asal sampai titik tujuan




**Gambar 15.1** Contoh *graph*

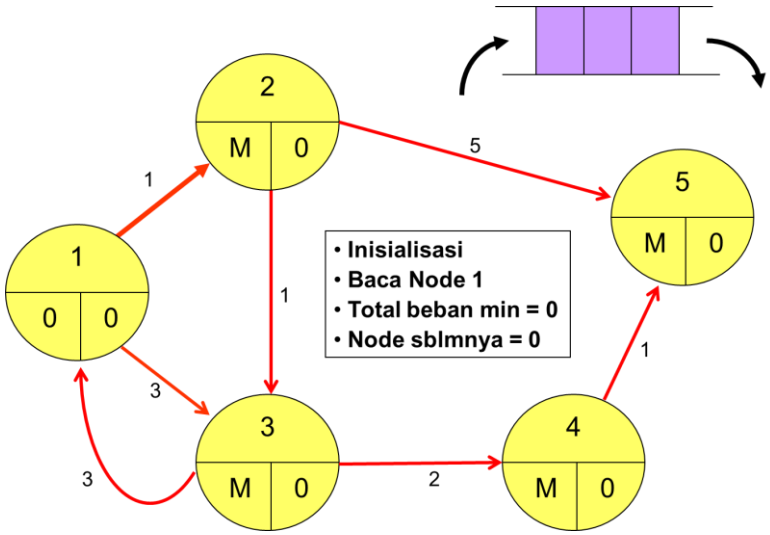
Sebagai ilustrasi, *graph* pada Gambar 15.1 di atas langkah-langkah nya sebagai berikut :

1. Tentukan titik asal = 1 dan titik tujuan = 5. Kita tentukan setiap node mempunyai nilai x sebagai nama node, nilai y sebagai beban minimal dan nilai z sebagai node sebelumnya seperti Gambar 15.2. Nilai y dan z disimpan sebagai vektor.



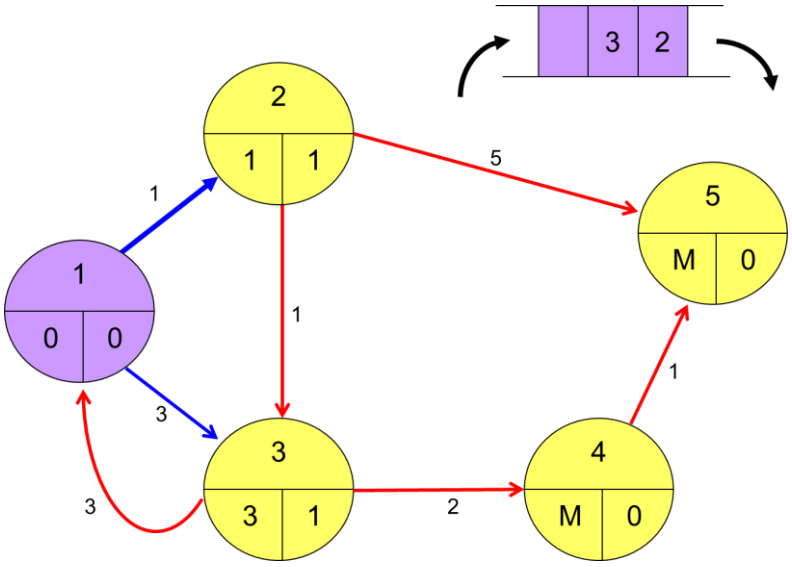

**Gambar 15.2** Nilai x, y dan z

2. Siapkan tumpukan untuk menyimpan node yang akan diproses. Sebagai inisialisasi awal, nilai y bernilai M (big integer) kecuali node awal bernilai 0, sedangkan nilai z bernilai 0. Node titik awal disimpan dalam tumpukan, dan baca nilai tersebut.



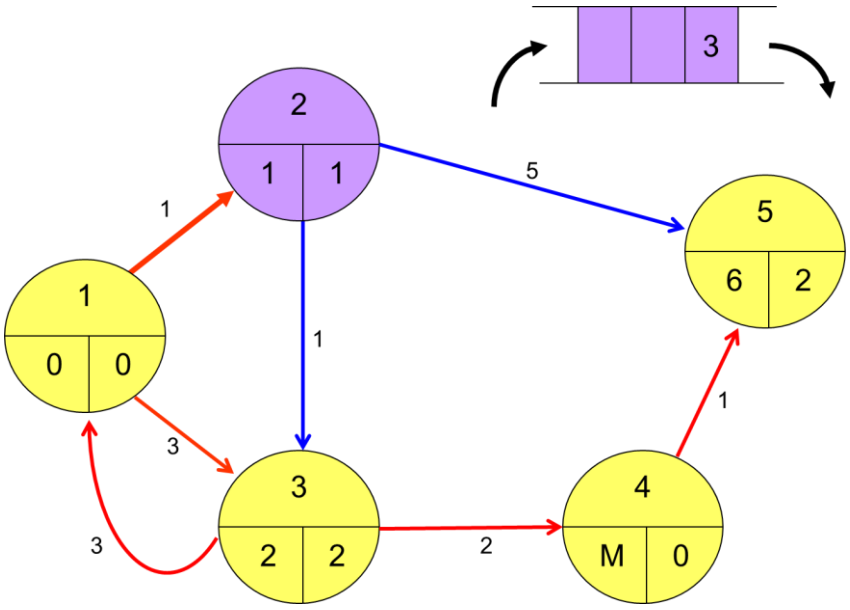

**Gambar 15.3** Inisialisasi awal

3. Baca node 1 dan lakukan perubahan nilai y dan z menjadi nilai beban dan node sebelumnya yang terkecil. Masukkan node z pada tumpukan jika node tersebut tidak ada di tumpukan, bukan titik awal dan bukan titik akhir.



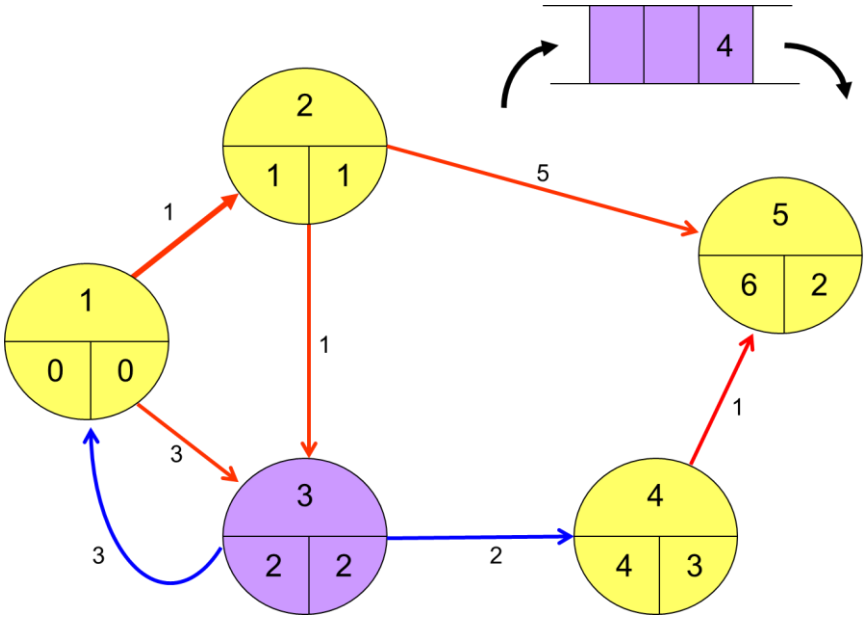

**Gambar 15.4** Baca node 1

4. Enqueue node pada tumpukan, baca node 2 dan lakukan hal yang sama dengan langkah 3.



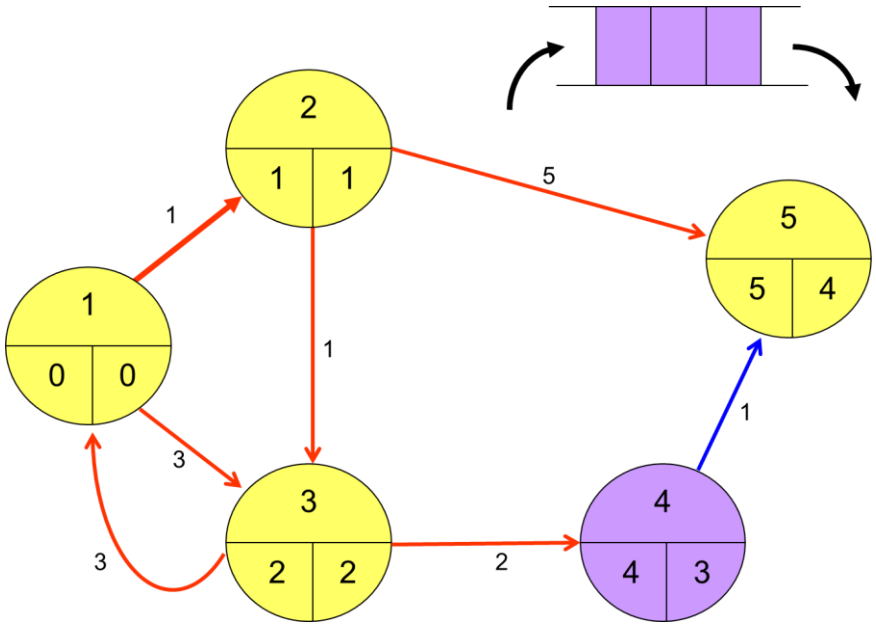

**Gambar 15.5** Baca node 2

5. Enqueue node pada tumpukan, baca node 3 dan lakukan hal yang sama dengan langkah 3.



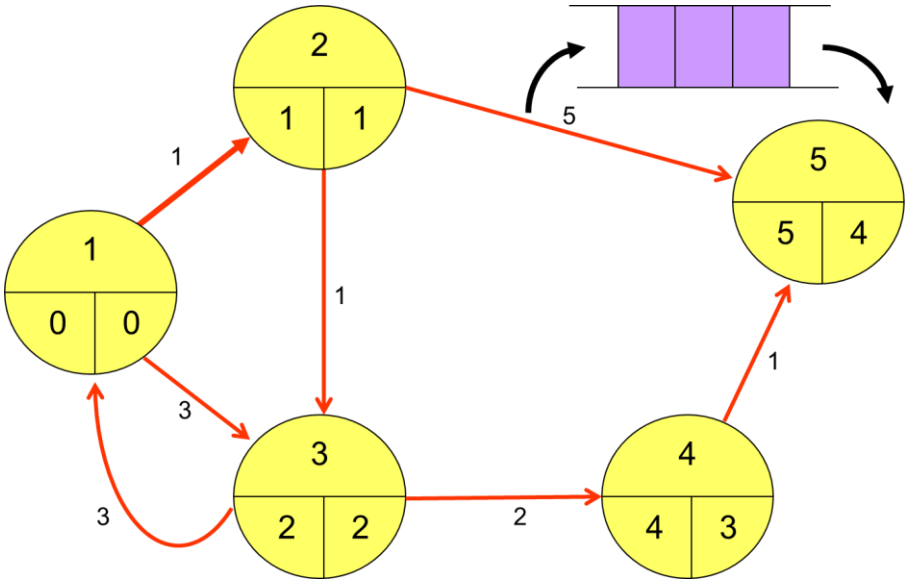

**Gambar 15.6** Baca node 3

6. Enqueue node pada tumpukan, baca node 4 dan lakukan hal yang sama dengan langkah 3.




**Gambar 15.7** Baca node 4

7. Bila tumpukan kosong, proses selesai sehingga menghasilkan vektor beban dan rute minimal.




**Gambar 15.8** Tumpukan kosong, proses selesai.

## C. TUGAS PENDAHULUAN

Jawablah pertanyaan berikut ini :

1. Tuliskan kembali algoritma antrian
2. Tuliskan algoritma djikstra untuk pencarian jalur terpendek pada Djikstra

## D. PERCOBAAN

### Percobaan 1 : Mendeklarasikan Graph serta inisialisasi Beban dan Rute

```
#include<stdio.h>

#define N 5
#define M 1000

void inisialisasi(int a, int Q[])
{
      for(int i=0; i<N; i++)
            if((i+1) == a)
                  Q[i]=0;
            else
                  Q[i]=M;
}

void Tampil(int data[N], char *judul)
{
      printf("%s = ",judul);
      for(int i=0; i<N; i++)
         if(data[i] >= M)
               printf("M ");
         else
               printf("%d ", data[i]);
      printf("\n");
}

void main()
{
      int input[N][N] = {M,1,3,M,M,
                         M,M,1,M,5,
                         3,M,M,2,M,
                         M,M,M,M,1,
                         M,M,M,M,M};

      int Beban[N], Rute[N]={0,0,0,0,0};
      int asal, tujuan;

      printf("Masukkan node asal : ");
      scanf("%d", &asal);
      printf("Masukkan node tujuan : ");
      scanf("%d", &tujuan);
      inisialisasi(asal, Beban);
      printf("Beban dan Rute awal\n");
      Tampil(Beban, "Beban");
      Tampil(Rute, "Rute");
}
```

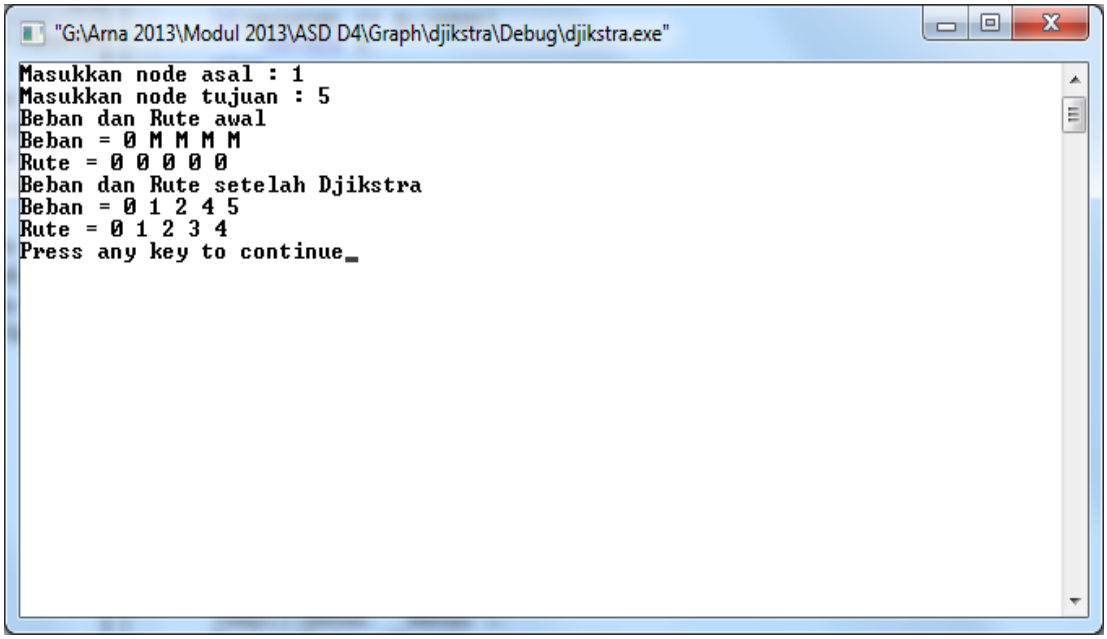
### Percobaan 2 : Mengimplementasikan Antrian

```
typedef struct{
      int item[N];
      int front;
      int rear;
      int count;
} Queue;

void inisialisasi_queue(Queue *q)
```

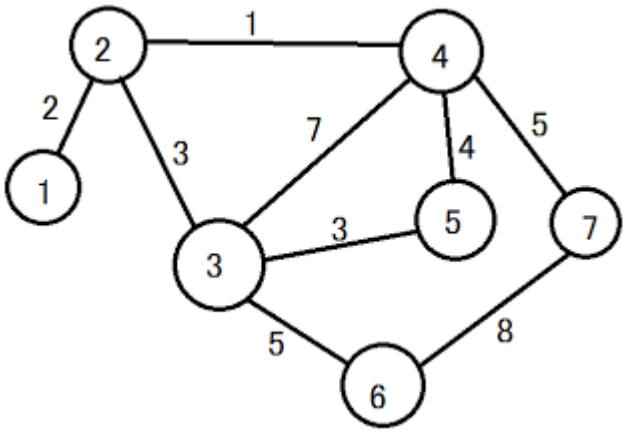
```
{
      q->front = q->rear = q->count = 0;
}

int Kosong(Queue *q)
{
      return q->count==0;
}

int Penuh(Queue *q)
{
      return q->count==N;
}

void Enqueue(Queue *q, int x)
{
      if(Penuh(q)){
            printf("Queue Penuh !\n");
            exit(1);
      }
      else {
            q->item[q->rear] = x;
            q->rear = (q->rear + 1) % N;
            q->count++;
      }
}

int Dequeue(Queue *q)
{
      int temp;
      if(Kosong(q)){
            printf("Queue kosong !\n");
            temp = -1;
      }
      else{
            temp = q->item[q->front];
            q->front = (q->front + 1) % N;
            q->count--;
      }
      return temp;
}
```

## E. LATIHAN

1. Tambahkan pada main program untuk mencari rute terpendek dengan algoritma Djikstra. Output berupa vektor Beban dan Rute setelah algoritma Djikstra seperti di bawah ini.




2. Tambahkan program untuk menampilkan rute terpendek dari vektor Rute dan Beban terkecil dari vektor Beban.
3. Running program beberapa kali dengan node asal dan tujuan berbeda dan analisa hasilnya.
4. Pada graph di bawah ini, selesaikan dengan algoritma djikstra. Jalankan program dan analisa hasilnya.




5. Buatlah studi kasus sebuah *graph* .Setelah itu jalankan program dan analisa hasilnya.

## F. LAPORAN RESMI

1. Kerjakan hasil percobaan(D) dan latihan(E) di atas dan tambahkan analisa.
2. Tuliskan kesimpulan dari percobaan dan latihan yang telah anda lakukan.