Penerapan Algoritma Program Dinamis untuk Persoalan Routing dalam Jaringan Komputer

Irvan Aditya (13510016)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13510016@std.stei.itb.ac.id

***Abstrak*—Salah satu perangkat yang memiliki peran penting dalam jaringan komputer adalah *router*. Salah satu fungsi r*outer adalah* menentukan jalur-jalur yang digunakan untuk menyampaikan paket data melalui *router* lainnya. Komunikasi antar *router* dalam penentuan jalur-jalur tersebut didefinisikan dalam protokol *routing*. Protokol *routing* yang paling banyak digunakan saat ini adalah OSPF dan RIP. Protokol OSPF dan RIP menggunakan dua bentuk algoritma program dinamis, yaitu algoritma Dijkstra dan algoritma Bellman-Ford.**

***Kata Kunci*—program dinamis, *routing*, OSPF, RIP, Dijkstra, Bellman-Ford**

I. PENDAHULUAN

Penggunaan jaringan komputer di dunia saat ini sudah sangat umum. Bahkan, jaringan komputer sudah mencakup seluruh dunia dan sudah menghubungkan sangat banyak komputer. Jaringan besar itu disebut internet. Jaringan internet bersifat terdesentralisasi. Setiap komputer di dalamnya bersifat independen.[1]

Internet, atau *internetwork* adalah kumpulan jaringan yang dihubungkan oleh perangkat-perangkat jaringan perantara yang dapat dilihat sebagai sebuah jaringan yang sangat besar. Sementara itu, *routing* adalah kegiatan menyampaikan informasi dalam internet dari sumber ke tujuan. Sepanjang perjalanan, informasi melewati paling sedikit satu perantara.[2]

Ada beberapa protokol yang digunakan untuk *routing*. *International Organization for Standardization* (ISO) mengembangkan seperangkat protokol standard untuk routing, yaitu *Open System Interconnection* (OSI). Di dalam OSI ada beberapa protokol routing untuk penggunaan yang berbeda-beda. Ada *Intermediate System-to-Intermediate System* (IS-IS), *End System-to-Intermediate System* (ES-IS) dan *Interdomain Routing Protocol* (IDRP).[3]

Namun, dalam kenyataannya ada dua cara routing yang paling sering digunakan, yaitu menggunakan algoritma *link state* dan menggunakan algoritma *distance vector*. Kedua algoritma tersebut didasarkan pada algoritma program dinamis. Algoritma *link state* digunakan untuk *Open Shortest Path First* (OSPF), sementara algoritma *distance vector* digunakan untuk *Routing Information Protocol* (RIP). Kedua algoritma *routing* tersebut menggunakan prinsip program dinamis.

Dalam makalah ini akan dibahas bagaimana implementasi algoritma program dinamis dalam persoalan *routing* dengan protokol OSPF dan RIP.

II. Dasar Teori Program Dinamis

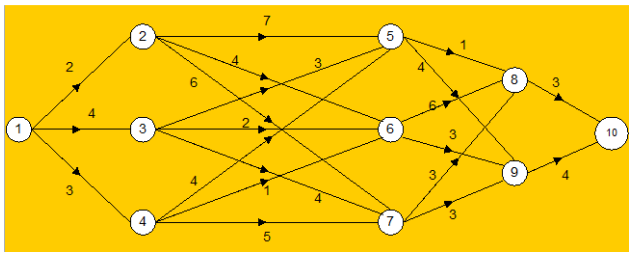
Program dinamis adalah metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan tahapan sedemikian sehingga solusi dapat dipandang sebagai serangkaian keputusan yang berkaitan.[4] Karakteristik persoalan yang dapat diselesaikan dengan program dinamis adalah:

1. Terdapat sejumlah berhingga pilihan yang mungkin.
2. Solusi pada setiap tahap dibangun dari hasil solusi tahap sebelumnya.
3. Ada persyaratan optimasi dan kendala untuk membatasi sejumlah pilihan yang harus dipertimbangkan pada suatu tahap.

Algoritma program dinamis serupa dengan algoritma *greedy*. Perbedaannya adalah dengan algoritma *greedy* hanya menghasilkan satu rangkaian keputusan, sementara algoritma program dinamis mempertimbangkan lebih dari satu rangkaian keputusan.

Dalam algoritma program dinamis dikenal Prinsip Optimalitas. Prinsip Optimalitas menyatakan bahwa jika solusi total optimal, bagian solusi hingga tahap ke-*k* juga optimal. Rangkaian keputusan yang optimal disusun berdasarkan prinsip tersebut. Hasil dari tahap ke-*k* dapat dipandang sebagai hasil optimal untuk memproses tahap ke-*k+1* tanpa harus kembali ke awal.

Program dinamis dapat dieksekusi dari depan atau dari belakang. Dari depan berarti algoritma berjalan dari asal menuju solusi. Dari belakang berarti algoritma berjalan dari solusi menuju asal.



Gambar 1 Contoh graf berbobot dan berarah

Misalnya untuk kasus lintasan terpendek dalam graf pada Gambar 1. Akan dicari lintasan terpendek dari simpul ke-1 menuju simpul ke-10. Ada empat tahap yang akan dilalui untuk mencapai solusi. Dalam setiap tahap, bobot graf dijumlahkan dengan bobot lintasan pada tahap sebelumnya.

Untuk penyelesaian lintasan terpendek dalam contoh graf pada Gambar 1 akan digunakan program dinamis dari depan, yaitu dari simpul ke-1 menuju simpul ke-10. Hasil perhitungan setiap dapat diperhatikan dalam Gambar 2 hingga Gambar 5.

s	Solusi Optimum	
	$f_1(s)$	x_1^*
2	2	1
3	4	1
4	3	1

Gambar 2 Hasil perhitungan tahap ke-1

x_2 / s	$f_2(x_2,s) = c_{x2,s} + f_1(x_2)$			Solusi Optimum	
	2	3	4	$f_2(s)$	x_2^*
5	9	7	7	7	3 atau 4
6	6	6	4	4	4
7	8	8	8	8	2, 3, 4

Gambar 3 Hasil perhitungan tahap ke-2

x_2 / s	$f_2(x_3, s) = c_{x3,s} + f_2(x_3)$			Solusi Optimum	
	5	6	7	$f_3(s)$	x_3^*
8	8	10	11	8	5
9	11	7	11	7	6

Gambar 4 Hasil perhitungan tahap ke-3

x_1 / s	$f_1(x_4, s) = c_{x4,s} + f_3(x_4)$		Solusi Optimum	
	8	9	$f_4(s)$	x_4^*
10	11	11	11	8 atau 9

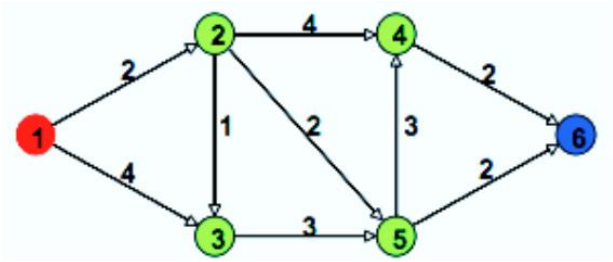
Gambar 5 Hasil perhitungan tahap ke-4

A. Algoritma Dijkstra

Algoritma Dijkstra adalah sebuah bentuk algoritma program dinamis. Algoritma ini cocok untuk graf berarah dan graf tidak berarah. Namun, ada keterbatasan untuk penggunaan algoritma Dijkstra, yaitu semua bobot graf tidak boleh negative dan graf harus terhubung.

Pseudo-code algoritma Dijkstra untuk kasus graf

berbobot dan berarah pada Gambar 6 dapat dilihat pada Gambar 7. *Pseudo-code* tersebut hanya menghasilkan total bobot terkecil dari kasus graf tersebut. Namun, dengan sedikit modifikasi pada *pseudo-code*, lintasan terpendek dapat dicari.[5]



Gambar 6 Contoh kasus graf berbobot dan berarah

```
dist[s] ←0                              (distance to source vertex is zero)
for all v ∈ V–{s}
    do dist[v] ←∞                       (set all other distances to infinity)
S←∅                                     (S, the set of visited vertices is initially empty)
Q←V                                     (Q, the queue initially contains all vertices)
while Q ≠∅                              (while the queue is not empty)
do  u ← mindistance(Q,dist)             (select the element of Q with the min. distance)
    S←S∪{u}                             (add u to list of visited vertices)
    for all v ∈ neighbors[u]
        do if  dist[v] > dist[u] + w(u, v)          (if new shortest path found)
           then    d[v] ←d[u] + w(u, v)             (set new value of shortest path)
                                                    (if desired, add traceback code)
return dist
```

Gambar 7 *Pseudo-code* algoritma Dijkstra

B. Algoritma Bellman-Ford

Algoritma Bellman-Ford serupa dengan algoritma Dijkstra. Algoritma Dijkstra tidak dapat menangani graf dengan bobot negatif. Namun, algoritmma Bellman-Ford ini dapat menangani graf dengan bobot negatif, bahkan dapat mendeteksi apakah ada lintasan berbobot negatif.[6]

Pseudo-code algoritma Bellman-Ford dapat dilihat pada Gambar 8.

```
Bellman-Ford ( G=(V,E,w), s )

1. For every vertex v
2.    d[v] = ∞
3. d[s]=0
4. For i=1 to |V|-1 do
5.    For every edge (u,v) in E do
6.       If d[v]>d[u]+w(u,v) then
7.          d[v]=d[u]+w(u,v), parent[v] = u

8.Return d[], parent[]
```

Gambar 8 *Pseudo-code* algoritma Bellman-Ford

III. PENERAPAN ALGORITMA PROGRAM DINAMIS DALAM PERSOALAN *ROUTING*

Dalam jaringan komputer, ada dua jenis algoritma *routing* yang paling popular, yaitu algoritma *link state* dan algoritma *distance vector*. Algoritma *link state* digunakan dalam protokol OSPF. Sedangkan algoritma *distance vector* digunakan dalam protokol RIP.

A. Penerapan Algoritma Program Dinamis dalam Protokol Routing *OSPF*

Router yang menggunakan protokol OSPF menggunakan algoritma *link state*. Algoritma *link state* didasarkan pada algoritma Dijkstra. Cara penghitungan bobot pada algoritma *link state* adalah dengan memanfaatkan status dari *router* dan *router* terdekatnya dalam jaringan.[7] Status yang dimaksud dalam hal ini adalah status koneksi dan jenis koneksi.[9] Status *router* tersebut selalu diperbarui dalam periode tertentu sehingga jika ada perubahan status *router* atau ada *router* yang terputus dari jaringan *router-router* terdekatnya akan langsung mengetahuinya. Karena itulah algoritma ini dinamakan algoritma link state.

B. Penerapan Algoritma Program Dinamis dalam Protokol Routing *RIP*

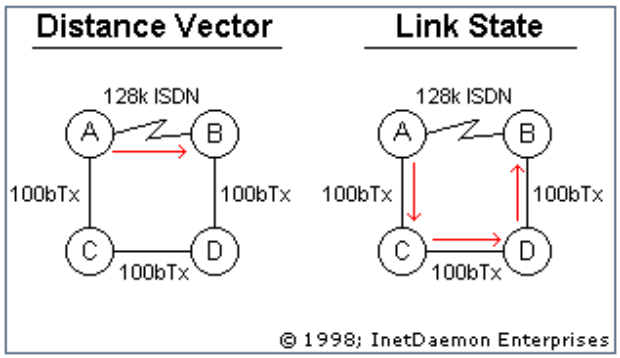
Router yang menggunakan protokol RIP menggunakan algoritma *distance vector*. Algoritma *distance vector* didasarkan pada algoritma Bellman-

Ford. Cara penghitungan bobot pada algoritma *distance vector* adalah dengan memanfaatkan perhitungan *hop* untuk memperhitungkan jarak antara asal dan tujuan.[8] *Hop* adalah jumlah simpul dalam graf jaringan yang dilalui dari asal ke tujuan.[9] Hasil perhitungan bobot tersebut kemudian akan disimpan dalam bentuk *routing table*. Sama seperti status *router* dalam algoritma link state, *routing table* dalam algoritma *distance vector* selalu diperbarui kepada *router-router* terdekatnya. *Routing table* ini yang akan dijadikan acuan untuk menentukan lintasan dalam pengiriman paket data setelahnya.

C. Perbandingan antara OSPF dan RIP

Kedua algoritma memiliki kelebihan dan kekurangannya tersendiri. Karena itu kedua protokol masih digunakan, tidak dihilangkan salah satunya.

Karena memanfaatkan status *router*, algoritma *link state* dapat memberikan kecepatan transfer data yang lebih baik. Pada Gambar 9 terlihat bahwa lintasan yang dipilih oleh algoritma *link state* lebih cepat melakukan transfer data walaupun jarak antar *router* lebih jauh.[9]



Gambar 9 Algoritma *link state* menghasilkan lintasan yang lebih cepat dalam transfer data

Karena algoritma *distance vector* menyimpan informasi lintasan dalam bentuk *routing table*, *router* dapat langsung mengirim paket ke tujuan tanpa perlu terlalu banyak *overhead*. Karena itu juga *router* tidak mengetahui bentuk topologi jaringan. Sedangkan algoritma *link state* menyimpan status *router-router* terdekat. Karena itu sebuah router dengan algoritma *link state* dapat menyusun lintasan terpendeknya sendiri tanpa membutuhkan bantuan *router* lain. Dengan kata lain, dapat dikatakan *router* mengetahui bentuk topologi jaringan. Dengan begitu algoritma *link state* lebih skalabel. Namun, karena itu algoritma *link state* membutuhkan lebih banyak *overhead* untuk mengirimkan data.[10]

V. KESIMPULAN

Algoritma program dinamis sudah lama digunakan dalam jaringan komputer. Algoritma tersebut digunakan untuk melakukan *routing*. Ada dua jenis algoritma untuk *routing* yang banyak digunakan, yaitu algoritma *link state* dan algoritma *distance vector*. Algoritma *link state* didasarkan pada algoritma Dijkstra, sedangkan algoritma *distance vector* didasarkan pada algoritma Bellman-Ford. Kedua algoritma tersebut merupakan algoritma program dinamis. Keduanya memiliki kelebihan dan kekurangan. Algoritma *link state* mengetahui topologi jaringan sehingga lebih skalabel. Algoritma *distance vector* membutuhkan lebih sedikit *overhead* dalam transfer data namun tidak mengetahui topologi jaringan.

REFERENSI

[1] http://www.webopedia.com/TERM/I/Internet.html diakses tanggal 17 Desember 2013.

[2] http://docwiki.cisco.com/wiki/Internetworking_Technology_Handbook diakses tanggal 17 Desember 2013.

[3] http://docwiki.cisco.com/wiki/Open_System_Interconnection_Routing_Protocol diakses tanggal 17 Desember 2013.

[4] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2013-2014/Program%20Dinamis%20(2013).ppt diakses tanggal 18 Desember 2013.

[5] http://math.mit.edu/~rothvoss/18.304.3PM/Presentations/1-Melissa.pdf/ diakses tanggal 17 Desember 2013.

[6] http://www.cs.rit.edu/~zjb/courses/800/lec15-1.pdf diakses tanggal 17 Desember 2013.

[7] http://docwiki.cisco.com/wiki/Open_Shortest_Path_First diakses tanggal 20 Desember 2013.

[8] http://docwiki.cisco.com/wiki/Routing_Information_Protocol diakses tanggal 20 Desember 2013.

[9] http://www.inetdaemon.com/tutorials/internet/ip/routing/dv_vs_ls.shtml diakses tanggal 20 Desember 2013.

[10] http://packetlife.net/blog/2008/oct/2/distance-vector-versus-link-state/ diakses tanggal 20 Desember 2013.

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

ttd

Irvan Aditya
13510016