Clustering Graf dengan Algoritma Rantai Markov

Theophilus Wellem[1], Yessica Nataliani[2]
[1,2]Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana
Jl. Diponegoro 52-60 Salatiga 50711, Jawa Tengah
Email: [1)]theophilus.wellem@staff.uksw.edu, [2)]yessica.nataliani@staff.uksw.edu

Abstract— Graph clustering is the task of grouping the vertices of the graph into clusters taking into consideration the edge structure of the graph in such a way that there should be many edges within each cluster and relatively few between the clusters. The objective of this paper is to apply Markov clustering algorithm. Two examples are used to demonstrate the algorithm. In the first example, simple graph is presented to illustrate the computation of this algorithm. While the second example is to study the hosts' interaction behavior using graph clustering algorithm. The Markov clustering algorithm is used to group (cluster) hosts which have interaction using the HTTP protocol. Using real network traces, the clustering results show that the algorithm successfully group the hosts to their corresponding clusters.
Keywords*:Clustering, clustering graf, algoritma rantai Markov, Traffic Dispersion Graph.*

I. PENDAHULUAN

Clustering adalah masalah pembelajaran unsupervised yang penting dan bermanfaat dalam menemukan struktur pada suatu data yang tidak mempunyai label. Tujuan dari clustering adalah untuk mengelompokkan objek ke dalam kelas-kelas atau beberapa cluster berdasarkan kesamaan karakteristiknya. Objek yang mempunyai karakter yang mirip akan dikelompokkan menjadi satu, sedangkan objek yang mempunyai karakter yang berbeda akan dikelompokkan ke dalam cluster yang berbeda. Oleh karena itu, suatu cluster merupakan kumpulan dari objek yang mirip satu sama lain, tetapi berbeda dengan cluster yang lain.

II. TEORI PENDUKUNG

2.1 Graf [1]

Graf G adalah pasangan himpunan (V, E), dimana V adalah himpunan berhingga dan tidak kosong yang disebut simpul (node, vertex) dan E adalah himpunan pasangan antar dua simpul yang disebut sisi (edge). Himpunan V adalah himpunan simpul, V(G) = V = {v1, v¬2, …, v¬n}, dimana n = |V| adalah jumlah simpul, yang sering disebut sebagai order dari graf G. Himpunan E adalah himpunan sisi, E(G) = E = {e1, e2, …, e¬m}, dimana m = |E| adalah jumlah sisi, yang sering disebut sebagai size dari graf G.

Jika simpul u dan v merupakan tetangga satu sama lain, maka simpul u (atau v) dan sisi e = uv dikatakan incident satu sama lain. Dua simpul u dan v dikatakan berdekatan (adjacent) jika ada sisi di antara simpul u dan v, atau ada sepasang uv □ E, dimana u, v □ V. Jika arah dari pasangan (u, v) diketahui, maka pasangan ini disebut sebagai sisi berarah dan graf seperti ini disebut sebagai graf berarah (digraf). Jika graf tidak mempunyai arah, maka graf disebut sebagai graf tidak berarah.

Sebuah multi-graph G terdiri dari himpunan simpul V yang berhingga dan tidak kosong serta himpunan sisi E dimana setiap dua simpul dari G terhubung dengan sebuah sisi (kemungkinan bisa tidak terhubung). Jika dua atau lebih sisi menghubungkan pasangan simpul yang sama, maka sisi ini disebut dengan sisi paralel. Dalam multi-graph, sebuah simpul dapat terhubung dengan simpul itu sendiri. Sisi yang menghubungkan simpul yang terhubung dengan dirinya sendiri disebut sebagai loop. Sebuah graf sederhana adalah graf tanpa sisi paralel dan loop.

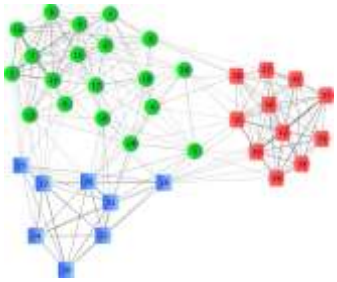
Matriks koneksi (adjacency matrix), adalah matriks dengan baris dan kolom yang diberi label oleh simpul graf, mempunyai elemen 0 atau 1 pada posisi (vi, vj), berdasarkan apakah vi dan vj adjacent atau tidak. Untuk graf tak berarah, matriks koneksi merupakan matriks simetris. Untuk graf sederhana, maka matriks koneksinya mempunyai nilai 0 pada diagonal utama. Didefinisikan adalah matriks koneksi dengan:

$$a_{ij} = \begin{cases} n, & \text{jika terdapat } n \text{ sisi dari } v_i \text{ ke } v_j \\ 1, & \text{jika terdapat satu sisi dari } v_i \text{ ke } v_j \\ 0, & \text{jika tidak terdapat } n \text{ sisi dari } v_i \text{ ke } v_j \end{cases}$$

Misalkan $G(V, w)$ adalah graf berarah yang berbobot dan berhingga, dengan $|V| = t$. Matriks asosiasi (*associated matrix*) G yang berada pada $\mathbb{R}^{txt}$, dilambangkan dengan M_G, didefinisikan dengan menetapkan entri $(M_G)_{ij}$ sama dengan $w(v_i, v_j)$. Jika diberikan matriks $M \in \mathbb{R}^{NxN}$, maka matriks asosiasi M, dilambangkan dengan ΓM, merupakan graf (V, w) dengan $|V| = N$ dan $w(v_i, v_j) = M_{ij}$.

2.2 Graf Clustering

Clustering graf berguna untuk mengelompokkan simpul pada graf ke dalam beberapa cluster dengan memperhatikan struktur sisi dari graf, sedemikian sehingga terdapat banyak sisi dalam suatu cluster dan relatif sedikit sisi di antara cluster [2]. Clustering graf merupakan hal yang penting dalam menemukan struktur dasar pada jaringan. Jaringan adalah kumpulan dari simpul (menggambarkan objek) yang terkoneksi dengan sisi (menggambarkan relasi antar objek). Jaringan sosial dapat dideskripsikan sebagai graf dimana seseorang direpresentasikan sebagai simpul dan hubungan pertemanannya direpresentasikan sebagai sisi. World Wide Web (WWW) juga dapat digambarkan sebagai graf, dimana halaman web direpresentasikan sebagai simpul dan link ke halaman lain dalam halaman web direpresentasikan sebagai sisi [3]. Gambar 1 menunjukkan contoh graf clustering.



Gambar 1. Contoh Graf Clustering

2.3 Rantai Markov

Rantai Markov digunakan untuk melakukan pemodelan matematika untuk memperkirakan kondisi di waktu yang akan datang berdasarkan kondisi di masa lalu. Jika diberikan kondisi saat ini, maka kondisi yang akan datang terpisah (independen) dari kondisi yang lalu. Setiap proses dapat mengubah kondisi saat ini menjadi kondisi yang lain, atau tetap pada kondisi yang sama, berdasarkan distribusi probabilitas tertentu.

Dimisalkan himpunan kondisi, S = {s1, s2, …, sn}. Proses dimulai dari suatu kondisi dan berubah ke kondisi yang lain. Perubahan kondisi pada sistem disebut sebagai transisi dan probabilitas yang berhubungan dengan perubahan kondisi disebut sebagai probabilitas transisi. Jika rantai saat ini berada pada kondisi si, maka kondisi tersebut akan bergerak ke kondisi sj pada langkah berikutnya, sehingga probabilitas transisinya dilambangkan dengan pij. Probabilitas ini tidak bergantung pada kondisi rantai tersebut sebelumnya. Proses ini juga dapat membuat suatu kondisi tetap berada pada kondisi yang sama [4].

Matriks transisi digunakan untuk menggambarkan transisi rantai Markov. Semua probabilitas transisi antar kondisi ditunjukkan dengan matriks transisi. Elemen dari setiap kolom pada matriks transisi merupakan bilangan positif dan totalnya adalah 1. Setiap elemen pij adalah probabilitas suatu kondisi berpindah dari kondisi i ke kondisi j.

2.4 Traffic Dispersion Graph (TDG)

TDG didefinisikan sebagai graf yang merepresentasikan berbagai interaksi pada simpul. Untuk trafik jaringan IP, simpul di TDG merupakan host dengan alamat IP yang unik, sedangkan sisi didefinisikan berdasarkan, misalnya, pertukaran paket antara dua host. Dengan kata lain, sisi merepresentasikan koneksi antara dua host tersebut. Koneksi dari sisi dapat didefinisikan dengan banyak cara, tergantung pada apa penyaring sisi yang digunakan, seperti yang sudah dibahas pada [5]. Pembahasan lebih lanjut tentang TDG dapat dilihat pada [5] dan [6].

III. ALGORITMA CLUSTERING DENGAN RANTAI MARKOV

Algoritma rantai Markov pertama kali diperkenalkan van Dongen [7]. Ide dari algoritma ini adalah untuk mensimulasikan aliran dalam graf, yang terdiri dari dua tujuan, yaitu untuk menguatkan aliran mana saat ini kuat dan untuk melemahkan aliran yang saat ini lemah. Dua operasi penting dalam algoritma ini adalah ekspansi dan inflasi. Ekspansi adalah operasi untuk mencari pangkat dari matriks transisi rantai Markov, sedangkan inflasi adalah sebuah operasi yang bertanggung jawab pada penguatan dan pelemahan suatu kondisi. Parameter inflasi (r) mengontrol penguatan atau pelemahan tersebut. Operator inflasi juga dapat didefinisikan sebagai berikut: Diberikan matriks , , dan bilangan real non negatif r. Matriks yang dihasilkan oleh penskalaan dari setiap kolom M dengan koefisien pangkat e dilambangkan dengan , dimana disebut sebagai operator inflasi dengan koefisien pangkat e. Secara formal, operasi didefinisikan sebagai:

$$(\Gamma_r M)^e{}_{pq} = \frac{(M_{pq})^e}{\sum_{i=1}^{k}(M_{iq})^e}$$

Jika nilai r tidak diketahui, maka diasumsikan r = 2.

Algoritma Clustering dengan Rantai Markov

Input: graf tak berarah, bilangan pangkat e, dan parameter inflasi r

1.Buat matriks asosiasi.

2.Tambahkan loop ke diri sendiri untuk setiap simpul.

(Hal ini dilakukan karena rantai Markov memungkinkan suatu kondisi untuk tetap pada kondisi semula, tidak berpindah ke kondisi yang lain).

3.Normalisasikan matriks atau buat matriks probabilitas transisi.

4.Lakukan proses ekspansi dengan memangkatkan matriks dengan bilangan pangkat e.

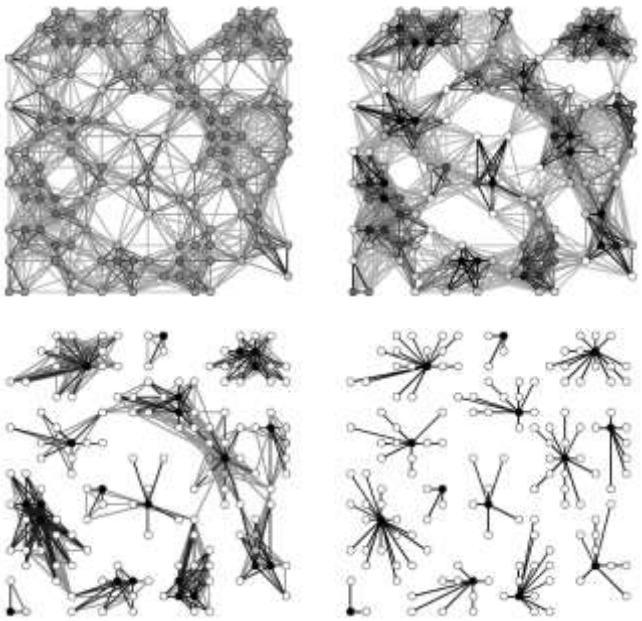
5.Lakukan proses inflasi dari matriks hasil Langkah 4 dengan parameter r.

6.Ulangi Langkah 5 dan 6 sampai konvergen.

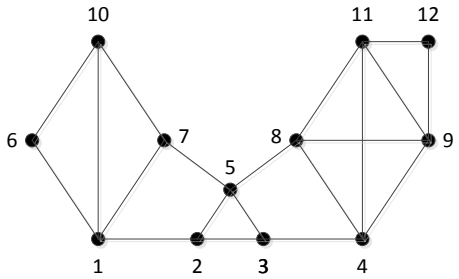
7.Interpretasikan matriks hasil untuk mendapatkan hasil clustering.

Output: data yang tercluster

Proses clustering dikatakan konvergen jika sudah tidak ada perubahan signifikan pada cluster baru. Gambar 2 menggambarkan contoh graf clustering dengan algoritma rantai Markov. Dari gambar tersebut dapat dilihat proses ekspansi dan inflasi. Jika diperhatikan, ada sisi-sisi baru yang terbentuk karena proses ekspansi. Selain itu, dapat dilihat juga ketebalan sisi yang berbeda-beda. Sisi dengan garis yang lebih tebal menunjukkan hubungan yang lebih kuat jika dibandingkan dengan sisi dengan garis yang lebih tipis. Lemah kuatnya hubungan tersebut terjadi karena proses inflasi.



Gambar 2. Contoh Graf Clustering dengan algoritma Rantai Markov [7]



Gambar 3. Graf untuk contoh 1

Untuk perhitungan, digunakan e = 2 and r = 2.

Langkah 1 dan 2: Bentuk matriks asosiasi (lihat Tabel 1) dan tambahkan loop ke diri sendiri untuk setiap simpul.

Tabel 1. Matriks asosiasi.setelah ditambahkan loop pada setiap simpul.

Simpul	1	2	3	4	5	6	7	8	9	10	11	12
1	1	1	0	0	0	1	1	0	0	1	0	0
2	1	1	1	0	1	0	0	0	0	0	0	0
3	0	1	1	1	1	0	0	0	0	0	0	0
4	0	0	1	1	0	0	0	1	1	0	1	0
5	0	1	1	0	1	0	1	1	0	0	0	0
6	1	0	0	0	0	1	0	0	0	1	0	0
7	1	0	0	0	1	0	1	0	0	1	0	0
8	0	0	0	1	1	0	0	1	1	0	1	0
9	0	0	0	1	0	0	0	1	1	0	1	1
10	1	0	0	0	0	1	1	0	0	1	0	0
11	0	0	0	1	0	0	0	1	1	0	1	1
12	0	0	0	0	0	0	0	0	1	0	1	1

Langkah 3: Normalisasikan matriks atau buat matriks probabilitas transisi (lihat Tabel 2).

Tabel 2. Matriks normalisasi.

+	1	2	3	4	5	6	7	8	9	10	11	12
1	0.2	0.25	0	0	0	0.33	0.25	0	0	0.25	0	0
2	0.2	0.25	0.25	0	0.2	0	0	0	0	0	0	0
3	0	0.25	0.25	0.2	0.2	0	0	0	0	0	0	0
4	0	0	0.25	0.2	0	0	0	0.2	0.2	0	0.2	0
5	0	0.25	0.25	0	0.2	0	0.25	0.2	0	0	0	0
6	0.2	0	0	0	0	0.33	0	0	0	0.25	0	0
7	0.2	0	0	0	0.2	0	0.25	0	0	0.25	0	0
8	0	0	0	0.2	0.2	0	0	0.2	0.2	0	0.2	0
9	0	0	0	0.2	0	0	0	0.2	0.2	0	0.2	0.33
10	0.2	0	0	0	0	0.33	0.25	0	0	0.25	0	0
11	0	0	0	0.2	0	0	0	0.2	0.2	0	0.2	0.33
12	0	0	0	0	0	0	0	0	0.2	0	0.2	0.33

Langkah 4: Proses ekspansi pada iterasi 1 dengan memangkatkan matriks dengan bilangan pangkat e =2, didapat M2 (lihat Tabel 3)

Tabel 3. Matriks hasil setelah proses ekspansi.

Simpul	1	2	3	4	5	6	7	8	9	10	11	12
1	0.257	0.113	0.063	0.000	0.100	0.261	0.175	0.000	0.000	0.258	0.000	0.000
2	0.090	0.225	0.175	0.050	0.140	0.067	0.100	0.040	0.000	0.050	0.000	0.000
3	0.050	0.175	0.225	0.090	0.140	0.000	0.050	0.080	0.040	0.000	0.040	0.000
4	0.000	0.063	0.113	0.210	0.090	0.000	0.000	0.160	0.160	0.000	0.160	0.133
5	0.100	0.175	0.175	0.090	0.230	0.000	0.113	0.080	0.040	0.063	0.040	0.000
6	0.157	0.050	0.000	0.000	0.000	0.261	0.113	0.000	0.000	0.196	0.000	0.000
7	0.140	0.100	0.050	0.000	0.090	0.150	0.225	0.040	0.000	0.175	0.000	0.000
8	0.000	0.050	0.100	0.160	0.080	0.000	0.050	0.200	0.160	0.000	0.160	0.133
9	0.000	0.000	0.050	0.160	0.040	0.000	0.000	0.160	0.227	0.000	0.227	0.244
10	0.207	0.050	0.000	0.000	0.050	0.261	0.175	0.000	0.000	0.258	0.000	0.000
11	0.000	0.000	0.050	0.160	0.040	0.000	0.000	0.160	0.227	0.000	0.227	0.244
12	0.000	0.000	0.000	0.080	0.000	0.000	0.000	0.080	0.147	0.000	0.147	0.244

Langkah 5: Proses inflasi pada iterasi 1 dengan parameter r =2, didapat (lihat Tabel 4)

Tabel 4. Matriks hasil setelah proses inflasi.

Simpul	1	2	3	4	5	6	7	8	9	10	11	12
1	0.380	0.087	0.027	0.000	0.077	0.295	0.201	0.000	0.000	0.320	0.000	0.000
2	0.047	0.347	0.210	0.017	0.150	0.019	0.066	0.011	0.000	0.012	0.000	0.000
3	0.014	0.210	0.347	0.055	0.150	0.000	0.016	0.046	0.009	0.000	0.009	0.000
4	0.000	0.027	0.087	0.302	0.062	0.000	0.000	0.184	0.143	0.000	0.143	0.083
5	0.058	0.210	0.210	0.055	0.406	0.000	0.083	0.046	0.009	0.019	0.009	0.000
6	0.142	0.017	0.000	0.000	0.000	0.295	0.083	0.000	0.000	0.184	0.000	0.000
7	0.113	0.069	0.017	0.000	0.062	0.097	0.333	0.011	0.000	0.147	0.000	0.000
8	0.000	0.017	0.069	0.175	0.049	0.000	0.016	0.287	0.143	0.000	0.143	0.083
9	0.000	0.000	0.017	0.175	0.012	0.000	0.000	0.184	0.288	0.000	0.288	0.278
10	0.246	0.017	0.000	0.000	0.019	0.295	0.201	0.000	0.000	0.320	0.000	0.000
11	0.000	0.000	0.017	0.175	0.012	0.000	0.000	0.184	0.288	0.000	0.288	0.278
12	0.000	0.000	0.000	0.044	0.000	0.000	0.000	0.046	0.120	0.000	0.120	0.278

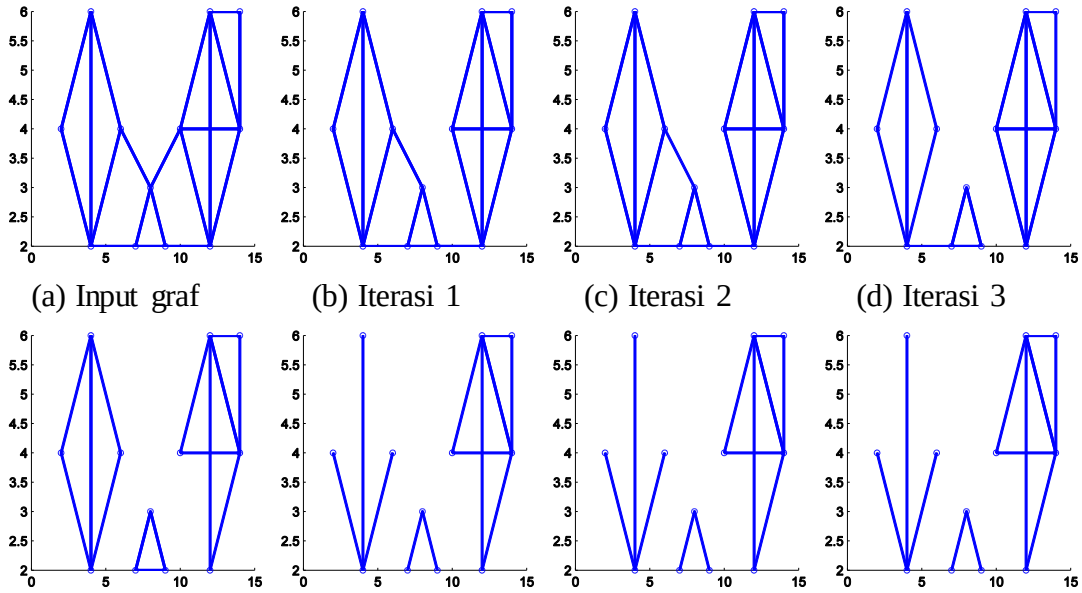
Ulangi langkah untuk proses ekspansi dan inflasi sampai konvergen, didapatkan hasil seperti pada Tabel 5

Tabel 5. Matriks hasil.

Simpul	1	2	3	4	5	6	7	8	9	10	11	12
1	1.000	0.000	0.000	0.000	0.000	1.000	1.000	0.000	0.000	1.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

5	0.000	1.000	1.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
6	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.500	0.000	0.000	0.000	0.500	0.500	0.000	0.500	0.500
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
11	0.000	0.000	0.000	0.500	0.000	0.000	0.000	0.500	0.500	0.000	0.500	0.500
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Dari Tabel 5, didapatkan tiga cluster, yaitu cluster untuk simpul 1, 6, 7, 10; cluster untuk simpul 2, 3, 5; dan cluster untuk simpul 4, 8, 9, 11, 12. Hasil graf clustering per iterasi ditunjukkan pada Gambar 4.



(a) Input graf (b) Iterasi 1 (c) Iterasi 2 (d) Iterasi 3

(e) Iterasi 4 (f) Iterasi 5 (g) Iterasi 6 (h) Iterasi 7 (output graf)

Gambar 4. Ilustrasi graf clustering

Contoh 2: Berikut merupakan aplikasi graf clustering pada jaringan komputer. TDG digunakan untuk merepresentasikan interaksi host, khususnya, komunikasi timbal balik HTTP antar host (host yang bertukar paket dengan source/destination TCP dengan nomor port 80). Umumnya, sisi-sisi pada TDG berarah karena simpul source dan destination diketahui dari informasi pada packet header. Namun, karena tulisan ini berfokus pada ada atau tidaknya komunikasi HTTP antar host, sehingga arah bisa diabaikan dan graf tidak berarah digunakan untuk merepresentasikan komunikasi tersebut. Simpul digunakan untuk merepresentasikan host dengan alamat IP yang unik, sedangkan sisi merupakan komunikasi host (traffic flows) berdasar pada source atau destination dengan nomor port 80.

Trace trafik jaringan yang digunakan dalam percobaan ini adalah trace real MAWI [8]. Durasi trace adalah 15 menit, dengan jumlah flow (alamat source IP dan alamat IP tujuan yang unik) adalah 408,043. Untuk mengkonstruksi TDG, flow dihasilkan (dan disaring menggunakan 80 sebagai source dan destination dari nomor port TCP) untuk setiap interval waktu (10 detik). Oleh karena itu, terdapat total 90 TDG yang dihasilkan. TDG untuk tiga interval pertama yang divisualisasikan menggunakan Graphviz [9] ditunjukkan pada Gambar 5, Gambar 6, Gambar 7. Tabel 6 menunjukkan properti trafik HTTP [10].

Tabel 6. Properti trafik HTTP (tiga interval pertama)

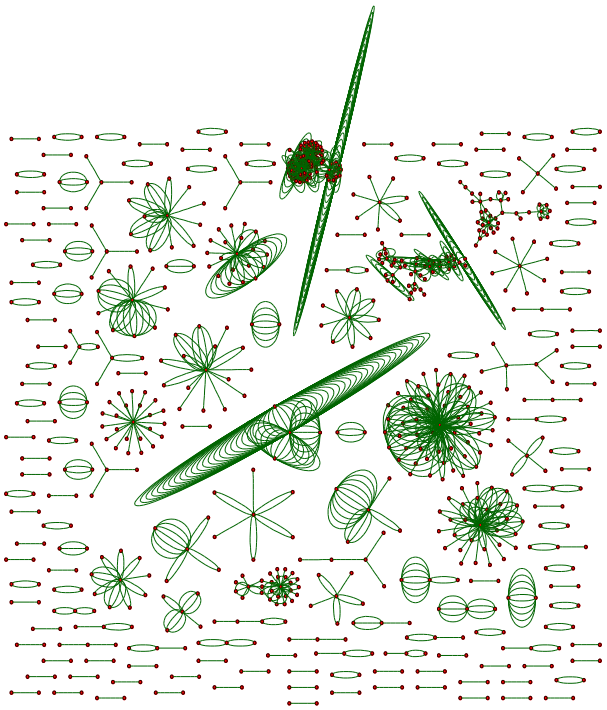
Interval	**Jumlah *source* IP**	**Jumlah *destination* IP**	**Jumlah IP yang unik (jumlah simpul)**	**Jumlah *flow* (jumlah sisi)**
Interval 1	478	539	791	1,413
Interval 2	435	536	771	1,405
Interval 3	436	483	726	1,324

Seperti yang telah disebutkan di atas, tiga interval pertama dari trace MAWI digunakan dan disaring menggunakan nomor port TCP 80 sebagai source atau destination. Pada interval 1, trace tersebut memiliki 1,413 komunikasi yang terbentuk dari 478 alamat source IP yang berbeda dan 539 alamat destination IP yang berbeda. Dari total 1,017 alamat IP, terdapat 791 alamat IP yang berbeda. Ini berarti bahwa interval 1 memiliki graf dengan 791 simpul dan 1,413 sisi untuk dicluster menggunakan algoritma rantai Markov. Setelah proses clustering, jumlah komunikasi (sisi) berkurang menjadi 806. Hal ini disebabkan karena beberapa komunikasi dengan source IP dan destination IP yang sama dikelompokkan ke dalam satu cluster. Terdapat juga beberapa komunikasi dari suatu source IP ke beberapa destination IP atau dari beberapa source IP ke suatu destination IP yang dikelompokkan ke dalam satu cluster yang sama. Dari interval pertama, 175 cluster terbentuk. Pada interval kedua dan ketiga, masing-masing terdapat 165 dan 150 cluster.

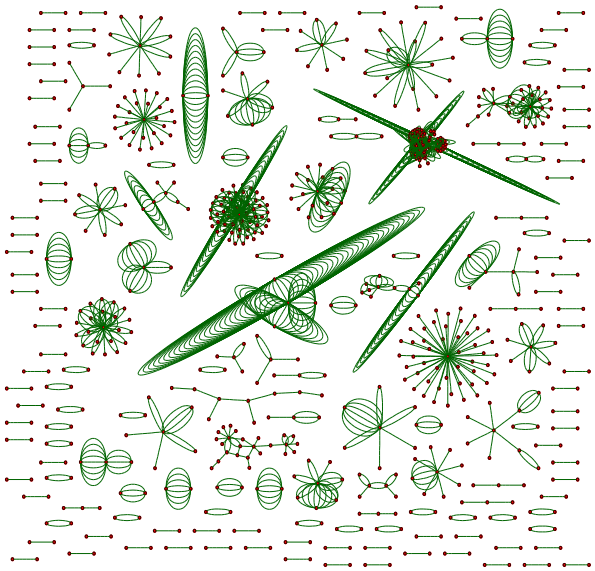
Tabel 7 menunjukkan hasil pengelompokan untuk setiap interval. Hasil dari graf clustering ditunjukkan pada Gambar 8, Gambar 9, dan Gambar 10.

Tabel 7. Hasil graf clustering dari trafik HTTP (tiga interval pertama)

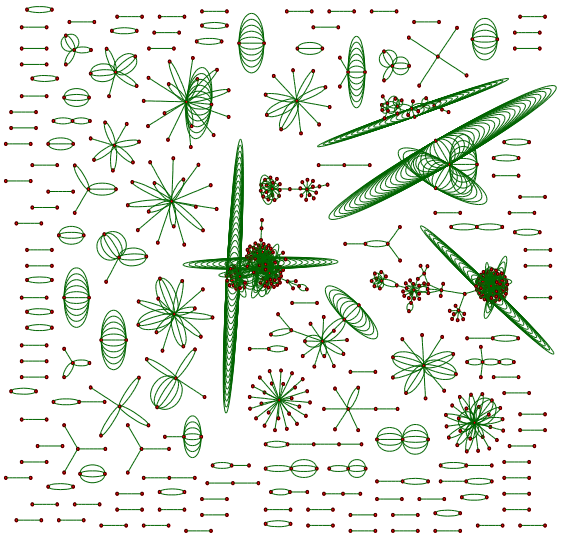
Interval	**Jumlah komunikasi (setelah clustering)**	**Jumlah cluster**
Interval 1	806	175
Interval 2	780	165
Interval 3	739	150



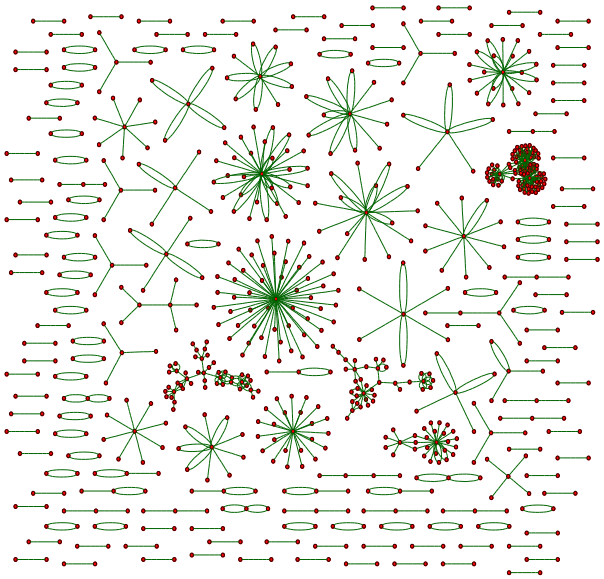
Gambar 5. Graf trafik HTTP untuk Interval 1 (10 detik)



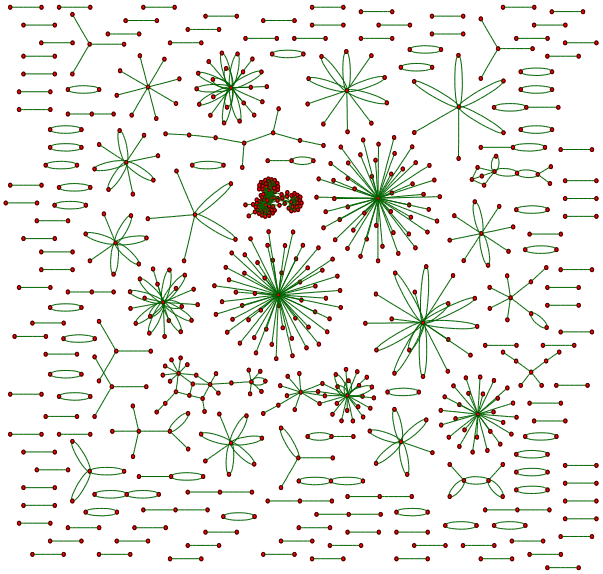
GAMBAR 6. GRAF TRAFIK HTTP UNTUK INTERVAL 2 (10 DETIK)



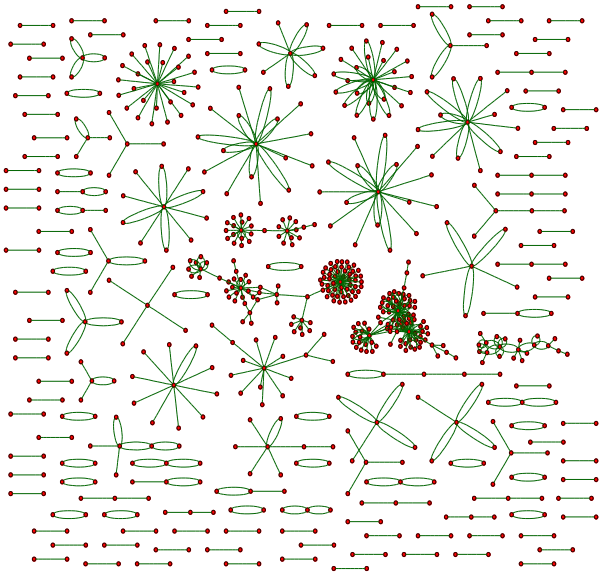
Gambar 7. Graf trafik HTTP untuk Interval 3 (10 detik)



Gambar 8. Graf trafik HTTP untuk Interval 1 (10 detik) setelah clustering



Gambar 9. Graf trafik HTTP untuk Interval 2 (10 detik) setelah clustering



Gambar 10. Graf trafik HTTP untuk Interval 3 (10 detik) setelah clustering

IV. KESIMPULAN

Tulisan ini menyajikan aplikasi algoritma clustering dengan rantai Markov, yang merupakan algoritma untuk clustering graf. Salah satu aplikasi yang dibahas adalah untuk mengelompokkan host sesuai dengan kesamaan perilaku mereka. TDG digunakan untuk mewakili komunikasi HTTP antara host. Hasil penelitian menunjukkan bahwa host yang sering berkomunikasi menggunakan HTTP berhasil dikelompokkan ke dalam satu cluster. Dengan menggunakan algoritma ini, komunikasi antara alamat IP suatu source ke alamat IP suatu destination, suatu source IP ke beberapa alamat destination IP yang berbeda, dan beberapa alamat source IP yang berbeda ke suatu alamat destination IP akan membuat alamat IP yang berbeda dari setiap kasus, dikelompokkan ke dalam satu cluster.

V. DAFTAR PUSTAKA

[1] Manongga, D., Nataliani, Y. (2015). Matematika Diskrit cetakan ke-2. Prenada Media (Kencana).

[2] Schaeffer, S. (2007). Graph clustering. Computer Science Review, 1, 27--64.

[3] Xu, X., Yuruk, N., Feng, Z. & Schweiger, T. A. J. (2007). SCAN: A Structural Clustering Algorithm for Networks. Proceedings of the 13th ACM SIGKDD International Conference on Knowledge <p> Discovery and Data Mining (p./pp. 824-833).

[4] Wilkinson, D. J. (2006). Stochastic modelling for systems biology. Boca Raton, Fla. [u.a.]: Chapman & Hall/CRC. ISBN: 978-1-584-88540-5.

[5] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese, "Network monitoring using tra c dispersion graphs (TDGs)," in Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, IMC '07, (New York, NY, USA), pp. 315-320, ACM, 2007.

[6] M. Iliofotou, "Exploring graph-based network traffic monitoring," in IEEE INFOCOM Workshops 2009, pp. 1-2, 2009.

[7] van Dongen, S. (2000). Graph Clustering by Flow Simulation. Unpublished doctoral dissertation, University of Utrecht.

[8] MAWI Working Group, "MAWI Working Group Traffic Archive." http://mawi.wide.ad.jp.

[9] "Graphviz: Open source graph visualization software." http://www.graphviz.org/.

[10] Nataliani, Y., Wellem, T. (2014) HTTP Traffic Graph Clustering using Markov Clustering Algorithm. International Journal of Computer Applications 90(2):37-41.