Algoritma Vertex Cover dan Aplikasinya

Kevin Winata /13510073
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
kevinwinata@students.itb.ac.id

***Abstract*— Dalam makalah ini akan dibahas mengenai permasalahan vertex cover dan aplikasinya. Vertex cover adalah sebuah himpunan simpul dari sebuah graf sederhana, dimana sisi-sisi dari graf tersebut bersisian dengan paling tidak 1 dari himpunan simpul tersebut. Aplikasi teori vertex cover yang akan dibahas antara lain : SNP Assembly Problem, dan Computer Network Security**

***Index Terms*— Teori Graf, Simpul, Sisi, Bersisian, Bertetangga, Vertex Cover, Minimum Vertex Cover, Algoritma, Kompleksitas, SNP Assembly, Computer Security**

I. PENDAHULUAN

Makalah ini akan membahas salah satu masalah yang terkenal dalam teori graf, yaitu vertex cover.

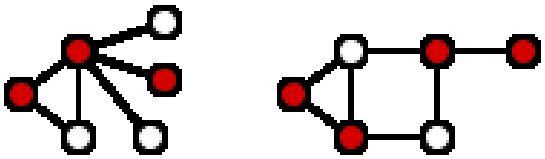
A. Graf

Graf adalah alat yang dipakai untuk memvisualisasikan hubungan antara objek-objek diskrit.
Sebuah graf memiliki sekumpulan simpul (vertex) dan sisi (edge), dan dinotasikan sebagai *G = (V, E)* dimana V adalah himpunan simpul-simpul, dan E adalah himpunan sisi-sisi dari graf G.
Graf sederhana adalah graf yang tidak mengandung gelang, yaitu adanya sisi yang menghubungkan sebuah simpul dengan dirinya sendiri, dan juga tidak mengandung sisi ganda, dimana ada lebih dari satu sisi yang menghubungkan dua simpul yang sama.
Terminologi pada graf :

1. Ketetanggaan (adjacency) yaitu simpul-simpul yang berhubungan langsung melalui suatu sisi.
2. Bersisian (incidency), sebuah sisi disebut bersisian dengan sebuah simpul apabila salah satu ujung dari sisi tersebut merupakan simpul yang dimaksud.
3. Derajat, yaitu besaran yang menyatakan jumlah sisi yang bersisian dari sebuah simpul.

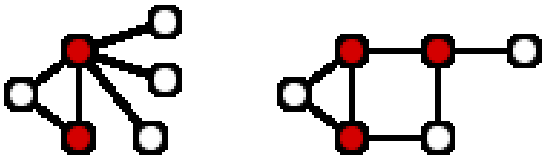
B. Vertex Cover

Vertex Cover dari sebuah graf sederhana adalah sebuah himpunan simpul dari graf, dimana semua sisi dari graf tersebut akan bersisian dengan paling tidak satu sudut yang berada dalam himpunan tersebut. Contohnya adalah :



1. Contoh Vertex Cover

Sedangkan minimum vertex cover adalah vertex cover yang memiliki jumlah simpul yang minimum. Untuk kedua graf contoh diatas, minimum vertex covernya adalah :



2. Contoh Minimum Vertex Cover

Kita bisa mengatakan sebuah simpul v dari graf G removable dari sebuah vertex cover C apabila C-v masih merupakan vertex cover dari graf G.

Notasi untuk himpunan simpul-simpul removable dari vertex cover C adalah ρ(C). Sebuah minimum vertex cover tidak boleh mengandung simpul removable.

Ada dua masalah utama yang berkaitan dengan vertex cover, yaitu pencarian minimum vertex cover dari sebuah graf, dan menentukan apakah ada suatu vertex cover dengan ukuran tertentu pada graf tersebut. Selama bertahun-tahun, berbagai algoritma telah dibuat untuk memecahkan berbagai persoalan terkait vertex cover. Salah satunya akan dibahas pada makalah ini, dan algoritma ini juga sekalian dapat menentukan ada tidaknya suatu vertex cover dengan ukuran tertentu dalam sebuah graf.

C. Kompleksitas Algoritma

Kemangkusan algoritma diukur dari waktu (*time*) eksekusi algoritma dan kebutuhan ruang (*space*) memori. Kemangkusan algoritma dapat digunakan untuk menilai algoritma yang bagus dari sejumlah algoritma penyelesaian masalah.
Besaran yang dipakai untuk menerangkan model abstrak

pengukuran waktu/ruang ini adalah kompleksitas algoritma. Ada dua macam kompleksitas algoritma, yaitu: kompleksitas waktu dan kompleksitas ruang.

- Kompleksitas waktu, $T(n)$, diukur dari jumlah tahapan komputasi yang dibutuhkan untuk menjalankan algoritma sebagai fungsi dari ukuran masukan n.
- Kompleksitas ruang, $S(n)$, diukur dari memori yang digunakan oleh struktur data yang terdapat di dalam algoritma sebagai fungsi dari ukuran masukan n.

Dalam makalah ini akan lebih dibahas mengenai kompleksitas waktu dibandingkan ruang, karena algoritma yang digunakan lebih menitikberatkan terhadap waktu dan langkah yang dibutuhkan daripada memori. Lebih lanjut, ruang memori yang dibutuhkan algoritma ini sebenarnya sangat kecil sehingga tidak perlu dipermasalahkan. Kompleksitas waktu sendiri dibagi menjadi 3 macam, yaitu Tmax, Tmin, dan Tavg yang masing-masing berarti kompleksitas waktu untuk kondisi worst case, best case, dan rata-ratanya.
Kompleksitas waktu asimptotik merupakan besaran yang digunakan untuk melihat pertumbuhan waktu yang diperlukan untuk menyelesaikan suatu algoritma terhadap jumlah elemen yang ingin diproses (n). Untuk kasus kompleksitas waktu berbentuk polinom, kompleksitas waktu asimptotiknya ditentukan berdasarkan orde terbesar.

II. ALGORITMA UNTUK MENCARI MINIMUM VERTEX COVER

Berikut akan dibahas salah satu algoritma yang dipakai dalam pencarian minimum vertex cover (dibuat oleh Ashay Dharwadker melalui bukunya *The Vertex Cover Algorithm*). Algoritma Dharwadker ini menurut saya cukup efisien dan memiliki kompleksitas waktu yang polinomial, sehingga akan dibahas disini.

A. Definisi Prosedur

❖ Prosedur 1 :
Input sebuah graf G dengan jumlah simpul n dan vertex cover dari G, yaitu C. Bila sudah tidak ada simpul removable dari C, output C. Sedangkan bila C masih mengandung simpul removable, untuk tiap simpul removable v kita mencari jumlah simpul removable $\rho(C-\{v\})$ dari vertex cover C. Kemudian kita berikan variabel v_{max} sehingga $\rho(C-\{v_{max}\})$ adalah maksimum, lalu kita tentukan $C-\{v_{max}\}$. Ulangi hingga vertex cover tidak punya simpul removable lagi.

❖ Prosedur 2 :
Input sebuah graf G dengan jumlah simpul n dan vertex cover dari G, yaitu C. Bila tidak ada simpul vdi C sehingga v mempunyai tepat satu tetangga, yaitu w diluar C, output C. Bila tidak, cari simpul v in C sehingga v mempunyai tepat satu tetangga, yaitu w diluar C. Definisikan $C^{v,w}$ dengan cara melepas v dari C dan memasukkan w ke C. Lakukan prosedur 1 pada $C^{v,w}$ dan output hasilnya.

B. Algoritma

Traversal i dari 1 sampai n, lalu lakukan hal-hal ini secara berurutan :

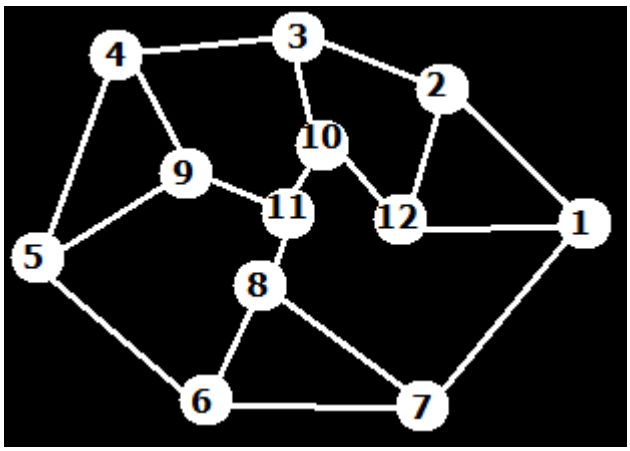
- Inisialisasi vertex cover $C_i = V-\{i\}$.
- Lakukan prosedur pertama pada C_i.
- Traversal r dari 1 sampai n-k, lakukan prosedur 2 sebanyak r kali.
- Hasilnya adalah minimum vertex cover C_i.

Lalu untuk setiap pasangan minimum vertex cover C_i, C_j yang didapat dari bagian pertama :

- Inisialisasi vertex cover $C_{i,j} = C_i \cup C_j$.
- Lakukan prosedur pertama pada C_i.
- Traversal r dari 1 sampai n-k, lakukan prosedur 2 sebanyak r kali.
- Hasilnya adalah minimum vertex cover $C_{i,j}$.

C. Contoh Penggunaan Algoritma

Kita pakai graf contoh sebagai berikut :



V = {1,2,3,4,5,6,7,8,9,10,11,12}
Kita akan mencoba mencari minimum vertex cover dengan jumlah maksimum 7.
Algoritma bagian 1 dimana kita memakai i=1 dan i=2 menghasilkan C1 dan C2 dengan jumlah 8. Karena itu kita menggunakan i = 3, lalu kita menginisialisasi $C_3 = V - \{3\} = \{1,2, 4,5,6,7,8,9,10,11,12\}$ (n = 11).

Simpul Removable dari C_3	Simpul Removable dari C_3-{v}	$\rho(C_3-\{v\})$
1	5,6,8,9,12	5
5	1,7,8,11,12	5
6	1,9,11,12	4
7	5,9,11,12	4

8	1,5,9,11	4
9	1,6,7,8,11	5
11	5,6,7,8,9,12	6
12	1,5,6,7,11	5

Maksimum $\rho(C_3-\{v\})= 6$ for v = 11. Hilangkan simpul 11 from C3.
Vertex Cover C3 = {1, 2, 4, 5, 6, 7, 8, 9, 10, 12}. (n = 10)

Simpul Removable dari C_3	Simpul Removable dari $C_3-\{v\}$	$\rho(C_3-\{v\})$
5	7,8,12	3
6	9,12	2
7	5,9,12	3
8	5,9	2
9	6,7,8	3
12	5,6,7	3

Maksimum $\rho(C_3-\{v\})= 3$ for v = 5. Hilangkan simpul 5 from C3.
Vertex Cover C_3 = {1, 2, 4, 6, 7, 8, 9, 10, 12}. (n = 9)

Simpul Removable dari C_3	Simpul Removable dari $C_3-\{v\}$	$\rho(C_3-\{v\})$
7	12	1
8	-	0
12	7	1

Maksimum $\rho(C_3-\{v\})= 1$ for v = 7. Hilangkan simpul 7 from C3.
Vertex Cover C_3 = {1, 2, 4, 6, 8, 9, 10, 12}. (n = 8)

Simpul Removable dari C_3	Simpul Removable dari $C_3-\{v\}$	$\rho(C_3-\{v\})$
12	-	0

Maksimum $\rho(C_3-\{v\})= 0$ for v = 12. Hilangkan simpul 12 from C3.
Kita mendapat Minimum Vertex Cover dari Graf G dengan (k = 7) = {1, 2, 4, 6, 8, 9, 10}, dan algoritma akan berhenti.

D. Kompleksitas

Kita akan mencoba menganalisis dan menghitung kompleksitas dari algoritma yang sudah diberikan.

Prosedur pertama :
Untuk mengecek sebuah simpul removable, dibutuhkan n^2 langkah, yaitu di tiap tetangganya yang berjumlah paling banyak n dibutuhkan n kali cek apakah simpul tersebut ada di vertex covernya. Kemudian untuk sebuah vertex cover, dibutuhkan n x $n^2 = n^3$ langkah untuk mencari ρ nya, dan n x $n^3 = n^4$ untuk mendapatkan simpul dimana ρ maksimum. Prosedur pertama ini berhenti ketika paling banyak n simpul telah dihapus, sehingga prosedur ini membutuhkan maksimum n^5 langkah.

Prosedur kedua :
Untuk mencari simpul yang mempunyai tepat 1 simpul tetangga, dibutuhkan paling banyak pemrosesan n simpul dan n kali untuk mencari apakah ada paling tidak 1 simpul yang diluar C. Kemudian prosedur menukar simpul, diikuti dengan prosedur 1 sehingga totalnya adalah $n^5 + n^2 + 1$.

Algoritma :
Bagian pertama men-loop n kali prosedur pertama dan prosedur kedua yang diulangi paling banyak n kali, sehingga menghasilkan $n\{n^5 + n(n^5 + n^2 + 1)\} = n^7 + n^6 + n^4 + n^2$. Kemudian bagian kedua membutuhkan n^2 pasangan vertex cover yang berbeda, sehingga menghasilkan $n^2\{n^5 + n(n^5 + n^2 + 1)\} = n^8 + n^7 + n^5 + n^3$. Karena bagian 1 dan 2 dilakukan secara berurutan, maka total algoritma membutuhkan tingkat kompleksitas $n^7 + n^6 + n^4 + n^2 + n^8 + n^7 + n^5 + n^3 = n^8 + 2n^7 + n^6 + n^5 + n^4 + n^3 + n^2$.

Keseluruhan dari perhitungan tadi adalah worst case (T_{max}). Karena kompleksitas algoritma ini berbentuk polinom, maka kompleksitas waktu asimptotiknya adalah $T(n) = O(n^8)$. Walaupun orde kompeksitas waktu asimptotiknya cukup besar, yaitu 8, algoritma ini masih cukup mangkus karena algoritma untuk menentukan minimum vertex cover memang sulit, dan jika dibandingkan dengan beberapa algoritma lain, algoritma ini mempunyai kompleksitas yang lebih sedikit.

Algoritma ini dapat digunakan untuk tiap graf sederhana dan akan selalu berhenti dengan kompleksitas yang polinomial, seperti telah dijabarkan di atas. Lebih jauh lagi, algoritma ini terbukti dapat menyelesaikan persoalan dari graf yang mempunyai simpul sebanyak n dan maksimum derajat simpul d dimana algoritma ini dapat mencari minimum vertex cover dengan ukuran paling besar $n - [n / (d + 1)]$.

III. APLIKASI DARI VERTEX COVER

A. SNP Assembly Problem

Dalam bidang biochemistry, banyak situasi dimana kita harus menyelesaikan konflik antar sekuens dengan cara mengecualikan beberapa sekuens tersebut. Kita bisa mendefinisikan graf konflik untuk sekuens-sekuens yang berkonflik satu sama lain.
Salah satu persoalan semacam ini adalah Single Nucleotide Polymorphism (SNP), yaitu sejenis mutasi dasar pada DNA. SNP merupakan sumber yang paling umum dari polimorfisme genetik dalam genom manusia.

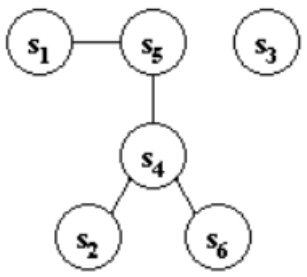
Definisi masalahnya :
SNP Assembly adalah tuple dengan 3 bagian, yaitu < S, F, R >, dimana S adalah himpunan dari SNP, dinotasikan $\{s_1 .. s_n\}$, kemudian F adalah himpunan dari fragmen berjumlah m, dinotasikan {f1 .. fm}, sedangkan R menyatakan relasi $R \rightarrow S \times F \{0, A, B\}$. Relasi ini menunjukkan apabila s_i tidak terjadi pada fragmen f_j, maka R bernilai 0, bila terjadi akan bernilai A atau B bergantung dari nilai s_i. 2 SNP dinyatakan berkonflik apabila terdapat 2 fragmen f1 dan f2 sehingga diantara R(si, fk), R(si, fl), R(sj, fk), R(sj, fl) tepat 3 diantaranya mengandung nilai tidak 0 yang sama (misalkan A) dan tepat 1 diantaranya bernilai kebalikan dari nilai tadi (misalkan B). Yang harus kita lakukan adalah mencari minimum vertex cover dari graf konflik yang menyatakan konflik di SNP Assembly tersebut. Mencari minimum vertex cover membuat kita jadi bisa menghilangkan sekuens secara minimal untuk menyelesaikan masalah SNP Assembly.

Contoh penyelesaian :
Melalui sebuah eksperimen, didapat tabel relasi R :

R	f_1	f_2	f_3	f_4	f_5
s_1	A	B			B
s_2	B	A	A	A	0
s_3	0	0	B	B	A
s_4	A	0	A	0	B
s_5	A	B	B	B	A
s_6	B		A	A	0

Dari tabel tersebut, kita bisa menggambar sebuah graf konflik.

- s_1 dan s_5 berkonflik karena $R(s_1, f_2) = B$, $R(s_1, f_5) = B$, $R(s_5, f_2) = B$, $R(s_5, f_5) = A$.
- s_4 dan s_5 berkonflik karena $R(s_4, f_1) = A$, $R(s_4, f_3) = A$, $R(s_5, f_1) = A$, $R(s_5, f_3) = B$.
- s_4 dan s_2 berkonflik karena $R(s_4, f_1) = A$, $R(s_4, f_3) = A$, $R(s_2, f_1) = B$, $R(s_2, f_3) = A$.
- s_4 dan s_6 berkonflik karena $R(s_4, f_1) = A$, $R(s_4, f_3) = A$, $R(s_6, f_1) = B$, $R(s_6, f_3) = A$.



Sisi-sisi pada graf konflik diatas menyatakan bahwa kedua simpul yang dihubungkan oleh sisi tersebut merupakan sekuens-sekuens yang berkonflik pada SNP Assembly Problem yang didefinisikan di atas.

Kemudian jika kita gunakan algoritma Dharwadker yang telah dijelaskan diatas, maka kita dapatkan minimum vertex covernya adalah $\{s_1, s_4\}$, atau alternatifnya adalah $\{s_4, s_5\}$. Dengan begitu solusi dari masalah SNP di atas adalah menghilangkan sekuens s1 dan s4 (atau s4 dan s5).

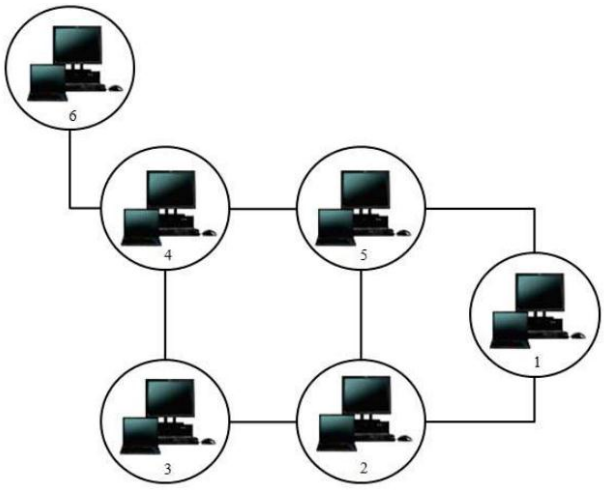
B. Computer Network Security

Vertex cover juga dapat diaplikasikan dalam pengamanan network. Worm, yang merupakan salah satu program malware yang dapat memperbanyak diri sendiri dan mengirimkannya melalui network, berkembang dengan cepat dan sejak tahun 2003-2004, worm sudah lebih low profile dan juga lebih membatasi penyebarannya sendiri, sehingga lebih sulit dideteksi.

Riset yang dilakukan oleh Eric Filiol, Edouard Franc, Alessandro Gubbioli, Benoit Moquet, Guillaume Roblot dalam paper-nya *Combinatorial Optimisation of Worm Propagationon an Unknown Network*, membahas tentang suatu jenis worm bernama combinatorial worm dan menggunakan vertex cover untuk mensimulasikan worm pada network yang besar dan menemukan cara yang optimal untuk mengamankan sistem dari worm tersebut secara real-time.

A. Combinatorial Worm

Combinatorian worm yang dipakai akan disimulasikan dengan kondisi sebagai berikut. Network benar-benar tidak diketahui oleh worm tersebut. Maksudnya, selain IP komputer pertama yang terinfeksi, IP lain yang tergabung dalam network yang sama tidak diketahui oleh attacker. Kemudian worm tersebut hanya berusaha untuk menginfeksi alamat IP yang benar-benar ada, bukan mencoba menginfeksi alamat IP yang didapat secara random. Dengan begitu worm tersebut akan meminimalkan jumlah koneksi yang diperlukan.
Nama combinatorial worm sendiri muncul karena taktik infeksinya yang memakai vertex cover. Idenya adalah sebagai berikut. Kita memakai fakta dimana terdapat hierarki koneksi antar komputer, sebagai contoh bila server A terhubung dengan server B dan server B terhubung dengan server C, server A tidak terhubung secara langsung dengan server C pada network tersebut. Bila worm tersebut mencoba menghubungkan server A dan server C secara langsung, kemungkinannya akan makin besar untuk terdeteksi. Oleh karena itu, attacker berusaha untuk dapat membuat model dari koneksi dengan graf secara progresif, dengan server/ alamat IP dimodelkan sebagai simpul, dan yang direpresentasikan sebagai sisi adalah apabila suatu ujung simpul sudah terinfeksi oleh ujung simpul yang lain. Solusinya adalah menggunakan minimum vertex cover untuk mengoptimalkan koneksi, dengan cara memilih server/ alamat IP yang memiliki koneksi paling banyak.



B. Network Security

Dengan mengerti cara kerja combinatorial worm, kita dapat mencari solusi untuk dapat mengamankan jaringan kita dari serangan worm-worm yang berprinsip sama, bukan hanya combinatorial worm saja. Dari simulasi cara penginfeksian worm yang telah dijelaskan di atas, maka riset dilanjutkan dengan tujuan menemukan cara yang optimal dalam mengamankan sistem network.

Untuk pengamanan, server-server yang termasuk dalam himpunan vertex cover memerlukan penanganan khusus. Penanganan yang semacam apa tidak akan dibahas disini, yang jelas pengidentifikasian server yang sedemikian akan mencegah infeksi worm dan menghapusnya dari network dengan lebih cepat dan lebih efisien. Secara umum, untuk semua jenis worm, membuat model graf untuk network yang dicurigai terinfeksi dan mencari vertex covernya akan sangat membantu dalam pertahanan terhadap worm, karena semua jenis worm pada dasarnya bereplikasi dan menggunakan koneksi dari komputer/server yang terinfeksi untuk mentransfer dirinya ke komputer/server lain.

IV. KESIMPULAN

- Algoritma Dharwadker merupakan algoritma yang dapat digunakan untuk menyelesaikan masalah mencari minimum vertex cover
- Algoritma Dharwadker memiliki kompleksitas waktu polinomial dengan kompleksitas waktu asimptotiknya adalah $O(n^8)$
- Algoritma vertex cover dapat menyelesaikan berbagai masalah yang dapat dimodelkan sebagai graf, sebagai contoh SNP Assembly Problem dan Computer Network Security

V. REFERENSI

[1] http://en.wikipedia.org/wiki/Vertex_cover, last access : 4.17 PM, 12/10/2011

[2] http://en.wikipedia.org/wiki/Graph_theory, last access : 4.17 PM, 12/10/2011

[3] Pirzada, Shariefuddin and Ashay Dharwadker, "*Applications Of Graph Theory*" Published in the Journal of The Korean Society for Industrial and Applied Mathematics (KSIAM), Vol. 11, No. 4, pp. 19-38, 2007.

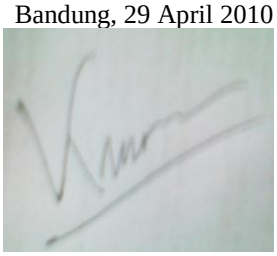
[4] Dharwadker, Ashay, *"The Vertex Cover Algorithm"*, http://www.dharwadker.org/vertex_cover, 2006.

[5] Filiol, Eric, Edouard Franc, Alessandro Gubbioli, Benoit Moquet, Guillaume Roblot, *"Combinatorial Optimisation of Worm Propagation on an Unknown Network"*, World Academy of Science, Engineering and Technology 34, 2007.

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, 29 April 2010



Kevin Winata / 13510073