Pemanfaatan Algoritma Semut untuk Penyelesaian Masalah Pewarnaan Graf

Anugrah Adeputra - 13505093

Program Studi Informatika, Sekolah Teknik Elektro & Informatika ITB
Jl. Ganesha No.10
If15093@students.if.itb.ac.id

ABSTRAK

Teori Graf merupakan salah satu topik bahasan yang menarik di bidang matematika dan ilmu komputer. Suatu graf merupakan kumpulan dari simpul-simpul/*nodes*) yang dihubungkan oleh sisi-sisi/*edges*. Representasi visual yang umum dikenal dari suatu graf adalah dengan menyatakan simpul sebagai titik dan sisi sebagai garis yang menghubungkan titik-titik yang ada.

Salah satu permasalahan yang cukup dikenal terkait dengan graf adalah Pewarnaan Graf. Pewarnaan Graf dapat diselesaikan dengan menggunakan berbagai jenis algoritma yang ada, salah satunya dengan menggunakan pemanfaatan Algoritma Semut. Algoritma Semut adalah algoritma yang meniru perilaku semut dan dapat digunakan untuk menyelesaikan berbagai permasalahan pada graf, di antaranya *Traveling Salesman Problem, , Job-Shop Scheduling,* Pewarnaan Graf, Permasalahan pada Jaringan, dan berbagai permasalahan terkait graf lainnya.

Pada makalah ini, akan dibahas mengenai pemanfaatan Algoritma Semut untuk menyelesaikan salah satu jenis permasalahan terkait graf, yakni mengenai Pewarnaan Graf.

Kata kunci: Graf, Simpul, Sisi, Pewarnaan Graf, Algoritma Semut

1. Pendahuluan

Algoritma Semut (*Ant Algorithm*) merupakan algoritma yang dimunculkan sebagai suatu pendekatan multi-agen (*Multi-agent approach*) terhadap optimasi berbagai permasalahan yang berkaitan dengan graf. Sampai saat ini, berbagai upaya pengembangan dilakukan untuk memperluas pemanfaatan dari Algoritma Semut. Berbagai pemanfaatan yang sudah umum digunakan antara lain untuk menyelesaikan permasalahan rute kendaraan, pengurutan sekuensial, pewarnaan graf, permasalahan *routing* pada jaringan dan berbagai pemanfaatan lainnya.

Algoritma Semut terinspirasi oleh pengamatan terhadap suatu koloni semut. Semut merupakan hewan yang hidup sebagai suatu kesatuan dalam koloninya dibandingkan jika dipandang sebagai individu yang hidup sendiri-sendiri dan tidak bergantung terhadap koloninya. Suatu perilaku penting dan menarik untuk ditinjau dari suatu koloni semut adalah perilaku mereka pada saat mencari makan, terutama bagaimana mereka mampu menentukan rute untuk menghubungkan antara sumber makanan dengan sarang mereka.

Ketika berjalan menuju sumber makanan dan sebaliknya, semut meninggalkan jejak berupa suatu zat yang disebut *Pheromone*. Semut-semut dapat mencium *Pheromone*, dan ketika memilih rute yang akan dilalui, semut akan memiliki kecenderungan untuk memilih rute yang memiliki tingkat konsentrasi *Pheromone* yang tinggi. Jejak *Pheromone* tersebut memungkinkan semut untuk menemukan jalan kembali ke sumber makanan atau sarangnya.

Seiring waktu, bagaimanapun juga jejak *Pheromone* akan menguap dan akan mengurangi kekuatan daya tariknya. Lebih lama seekor semut pulang pergi melalui suatu jalur, lebih tinggi pula jumlah *Pheromone* yang menguap. Sebagai perbandingan, sebuah jalur yang pendek akan diikuti oleh semut lainnya dengan lebih cepat, dan dengan demikian konsentrasi *Pheromone* akan tetap tinggi.

Penguapan *Pheromone* juga mempunyai keuntungan untuk mencegah konvergensi pada penyelesaian optimal secara lokal. Jika tidak ada penguapan sama sekali, jalur yang dipilih semut pertama akan cenderung menarik secara berlebihan terhadap semut-semut yang mengikutinya. Pada kasus yang demikian, eksplorasi ruang penyelesaian akan terbatasi.

Oleh karena itu, ketika seekor semut menemukan jalur yang bagus (jalur yang pendek) dari koloni ke sumber makanan, semut lainnya akan mengikuti jalur tersebut, dan akhirnya semua semut akan mengikuti sebuah jalur tunggal. Ide algoritma koloni semut adalah untuk meniru perilaku ini melalui 'semut tiruan' berjalan seputar grafik yang menunjukkan masalah yang harus diselesaikan.

Perilaku mengikuti jejak *Pheromone* tersebut telah dibuktikan secara eksperimental, digunakan oleh koloni semut untuk mengetahui rute terpendek untuk mencapai sarang atau sumber makanan berdasarkan jejak-jejak *Pheromone* yang ditinggalkan oleh masing-masing semut yang ada.

Berdasarkan perilaku tersebut, maka dikembangkanlah suatu algoritma untuk menyelesaikan suatu masalah komputasi dengan menemukan jalur terbaik melalui grafik.

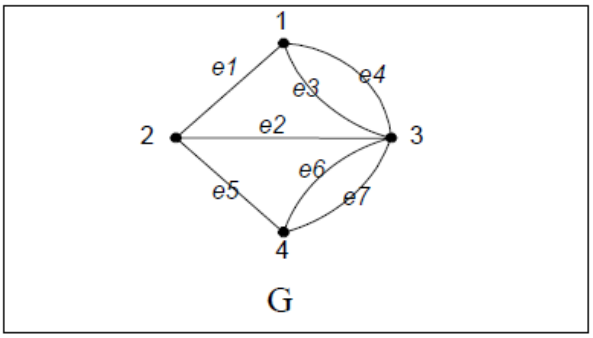
2. Tinjauan Pustaka

2.1. Graf

Graf digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut.

Graf $G=(V,E)$ didefinisikan sebagai berikut:

- V = Himpunan tidak kosong dari simpul-simpul (*vertices*) = $\{v_1,v_2,v_3,...v_n\}$
- E = Himpunan sisi-sisi yang menghubungkan masing-masing sepasang simpul = $\{e_1,e_2,e_3,...e_n\}$



Gambar 1

Gambar di atas menunjukkan sebuah graf G dengan:

- V = $\{1,2,3,4\}$
- E = $\{(1,2), (2,3) ,(1,3), (1,3), (2,4), (3,4), (3,4)\}$ = $\{e_1,e_2,e_3,e_4,e_5,e_6,e_7\}$

Jenis-Jenis Graf

- Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, maka graf digolongkan menjadi dua jenis:

1. **Graf sederhana** (*simple graph*).

Graf yang tidak mengandung gelang maupun sisi-ganda dinamakan graf sederhana. G_1 pada Gambar 2 adalah contoh graf sederhana

2. **Graf tak-sederhana** (*unsimple-graph*).

Graf yang mengandung sisi ganda atau gelang dinamakan graf tak-sederhana (*unsimple graph*). G_2 dan G_3 pada Gambar 2 adalah contoh graf tak-sederhana

- Berdasarkan jumlah simpul pada suatu graf, maka secara umum graf dapat digolongkan menjadi dua jenis:

1. **Graf berhingga** (*limited graph*)

Graf berhingga adalah graf yang jumlah simpulnya, *n*, berhingga.

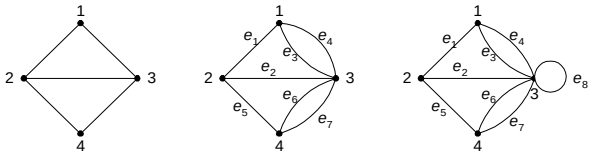
2. **Graf tak-berhingga** (*unlimited graph*)

Graf yang jumlah simpulnya, *n*, tidak berhingga banyaknya disebut **graf tak-berhingga**.

- Berdasarkan orientasi arah pada sisi, maka secara umum graf dibedakan atas 2 jenis:

1. **Graf tak-berarah** (*undirected graph*)

Graf yang sisinya tidak mempunyai orientasi arah disebut graf tak-berarah. Tiga buah graf pada Gambar 2 berikut adalah graf tak-berarah.

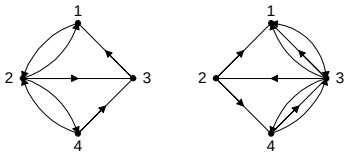


Gambar 2

Gambar graf paling kiri merupakan graf sederhana. Gambar graf tengah merupakan gambar graf ganda (karena memiliki sisi ganda), sedangkan gambar graf paling kanan merupakan graf semu karena memiliki gelang atau kalang, yakni sisi yang berawal dan berakhir pada simpul yang sama

2. **Graf berarah** (*directed graph* atau *digraph*)

Graf yang setiap sisinya diberikan orientasi arah disebut sebagai graf berarah. Dua buah graf pada Gambar 3 berikut adalah graf berarah.



Gambar 3

Graf kiri pada gambar di atas merupakan graf berarah, sedangkan graf kanan pada gambar di atas merupakan graf ganda berarah.

Berikut merupakan tabel yang merekapitulasi sifat-sifat dari masing-masing jenis graf yang telah dijelaskan di atas:

Jenis	**Sisi**	Sisi ganda?	Sisi gelang?
Sederhana	Tak-berarah	Tidak	Tidak
Ganda	Tak-berarah	Ya	Tidak
Semu	Tak-berarah	Ya	Ya
Berarah	Bearah	Tidak	Ya
Ganda Berarah	Bearah	Ya	Ya

Tabel 1 Jenis-jenis graf [ROS99]

Permasalahan Pewarnaan Graf

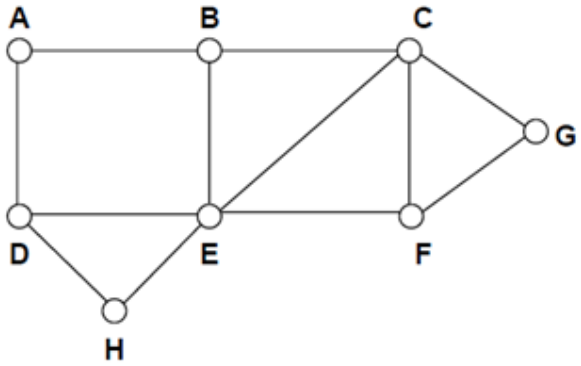
Permasalahan Pewarnaan Graf merupakan sebuah permasalahan optimasi yang sudah dikenal luas yang termasuk dalam kelas permasalahan tipe penetapan (*Assignment Type Problem*)

Pewarnaan Graf adalah proses pemberian warna terhadap simpul sedemikian sehingga dua simpul yang berdampingan (bersisian) tidak memiliki warna yang sama.

Jumlah minimum warna yang dibutuhkan disebut Bilangan Kromatis dari *G*, ditulis K(*G*). Dalam memecahkan Problema Pewarnaan, kita selalu berusaha mewarnai simpul menggunakan banyak warna minimal.

Salah satu algoritma yang cukup dikenal secara luas untuk menangani Permasalahan Pewarnaan Graf adalah Algoritma Welch-Powell. Berikut merupakan penjabaran dari Algoritma Welch-Powell tersebut:

1. Urutkan semua simpul berdasarkan derajatnya, dari derajar besar ke derajat kecil.
2. Ambil warna pertama (misalnya merah), warnai simpul pertama yang sudah kita urutkan berdasarkan derajatnya tadi. Kemudian warnai simpul berikutnya yang tidak berdampingan dengan simpul pertama tadi dengan warna yang masih sama (merah).
3. Kemudian kita lanjutkan dengan warna kedua, dan seterusnya, sampai semua simpul telah diberi warna.

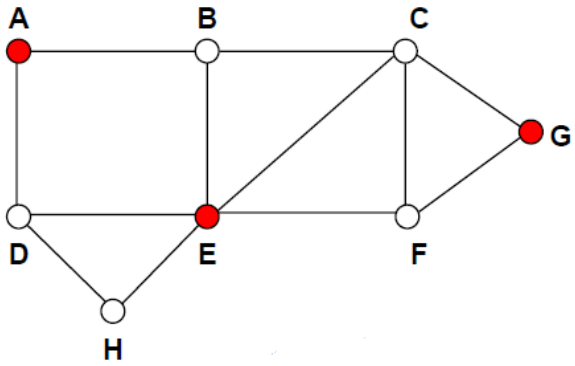


Gambar 4

Urutkan simpul berdasarkan derajatnya dari besar ke kecil: Simpul berderajat terbesar adalah E, yaitu 5 (mempunyai 5 ruas) kemudian simpul C berderajat 4, B,D,F masing-masing berderajat 3 dan A,H,G masing-masing berderajat 2.

Jadi Urutannya adalah : **E,C,B,D,F,A,H,G**

Ambil warna pertama, misalnya Merah. Beri warna Merah simpul E (karena E adalah simpul urutan pertama).Kemudian cari simpul yang ***tidak berdampingan*** dengan simpul E, beri warna yang sama (merah)

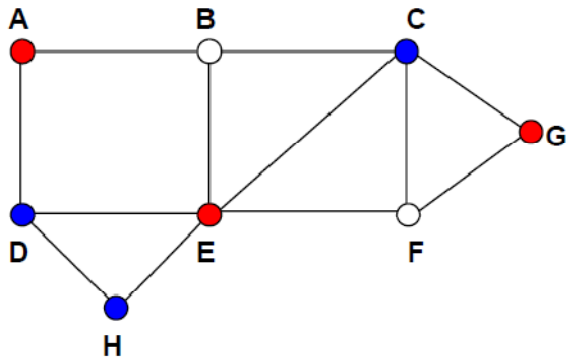


Gambar 5

Sehingga didapat urutan simpul yang belum diberi warna sbb : **C,B,D,F,H**

Ambil warna kedua, misalnya Biru, kemudian warnai simpul C (karena simpul C sekarang ada diurutan pertama). Kemudian cari simpul yang ***tidak berdampingan*** dengan simpul C, beri warna yang sama (Biru).

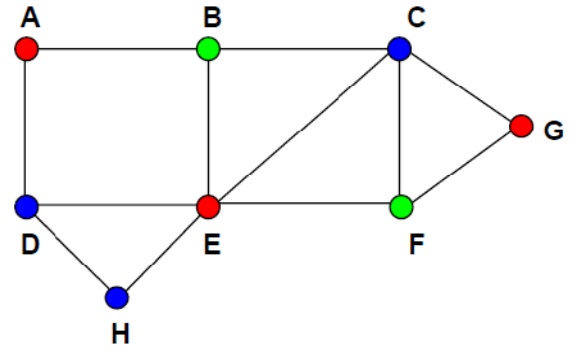
Kita berikan warna yang sama pada simpul D dan H dengan warna simpul C yaitu biru karena Simpul D dan H tidak berdampingan dengan simpul C



Gambar 6

Sehingga didapat urutan simpul yang belum diberi warna sbb : **B,F**

Ambil warna ketiga, misalnya Hijau, warnai simpul B dan F.



Gambar 7

Pewarnaan telah selesai, Graf merupakan Graf berwarna 3. Jadi bilangan kromatik dari graf G = K(G) = 3.

2.2. Algoritma Semut (*Ant Algorithm*)

Sejarah Algoritma Semut

Pada tahun 1996, dunia AI pun ikut belajar dari semut dengan diperkenalkannya algoritma semut, atau *Ant Colony Optimization*, sebagai sebuah simulasi multi agen yang menggunakan metafora alami semut untuk menyelesaikan *problem* ruang fisik.

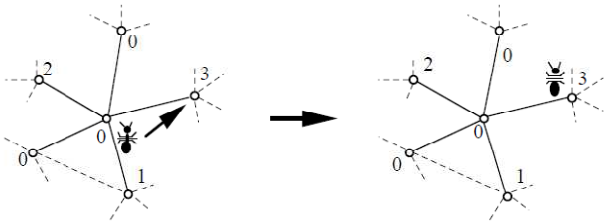
Algoritma semut diperkenalkan oleh **Moyson** dan **Manderick** dan secara meluas dikembangkan oleh **Marco Dorigo**, merupakan teknik probabilistik untuk menyelesaikan masalah komputasi dengan menemukan jalur terbaik melalui grafik. Algoritma ini terinspirasi oleh perilaku semut dalam menemukan jalur dari koloninya menuju makanan.

3. Isi dan Pembahasan

Algoritma Semut untuk Pewarnaan Graf

Algoritma yang dimanfaatkan untuk pewarnaan graf adalah merupakan suatu sistem multi agen yang berbasis pada pencarian paralel (*Parallel Search*). Dalam algoritma ini, sejumlah "semut" berkeliling ke simpul-simpul yang ada pada graf dan mewarnai setiap simpul yang dikunjungi berdasarkan kepada kriteria lokal tertentu.

Pada iterasi-iterasi yang diberikan, setiap "semut" berjalan dari simpul yang ditempatinya ke simpul tetangga yang memiliki nilai pelanggaran maksimum di antara tetangga-tetangga simpul tersebut dan mengganti warna yang ada dengan warna lain dengan tujuan untuk meminimalisasi nilai pelanggaran dari simpul tersebut. Nilai pelanggaran untuk suatu simpul *i* dikomputasi sebagai jumlah simpul tetangga yang memiliki warna yang sama dengan *i*. Perilaku di atas diilustrasikan pada gambar berikut:



Gambar 8

Keterangan: Angka pada setiap simpul menyatakan nilai pelanggaran dari masing-masing simpul.

Perilaku berikut diulangi secara acak untuk semua "semut": "semut " akan bergerak menuju simpul tetangga yang terburuk dengan suatu probabilitas tertentu p_n (jika tidak, maka ia akan bergerak secara acak ke simpul tetangga manapun) dan memberikan suatu warna yang memiliki probabilitas tertentu p_c (jika tidak, maka warna akan ditetapkan secara acak). Berdasarkan sifat probabilistik dari algoritma tersebut, memungkinkan "semut" untuk menghindari nilai minimum lokal dan mendapatkan batas mendekati nilai minimum global.

Proses yang dilakukan secara simultan oleh sekumpulan "semut" ini diulangi hingga mencapai hasil yang optimal.

Pemanfaatan Algoritma Semut untuk Pewarnaan Graf dapat diformulasikan sebagai berikut:

.

Misalkan sebuah graf $G = (V,E)$ di mana $V = (v_1,...v_n)$ adalah kumpulan simpul dan $E = (e_1,...e_n)$ adalah kumpulan sisi. Pewarnaan-q dari suatu graf adalah suatu pemetaan $c:V \rightarrow \{1,2,.....q\}$ sehingga $c(v_i) \neq c(v_j)$ dengan v_i dan v_j dihubungkan oleh sebuah sisi $<v_i, v_j> \in E$. Pewarnaan yang optimal dari sebuah graf G adalah pewarnaan-q dengan nilai q sekecil mungkin. Nilai minimal dari q disebut bilangan kromatik dengan simbol $\chi(G)$.

$$x_i^j = \begin{cases} 1, & \text{if vertex } i \text{ has color } j \\ 0, & \text{otherwise} \end{cases}$$

X Set of variables x, $X = \{x_1^1, x_1^2, ..., x_1^n, x_2^1, ..., x_n^n\}$.

J_i Set of colors admissible for item i $(1 \le i \le n)$, $J_i = \{1,...n\}$.

$$pgt(x) = \begin{cases} 1, & \text{if } x>0 \\ 0, & \text{otherwise} \end{cases}$$

Himpunan X memberikan penetapan warna ke seluruh simpul. Selalu memungkinkan untuk mewarnai suatu graf $G = (V,E)$ dalam $n = |V|$ warna, sehingga himpunan X memiliki nxn variable x. Fungsi Objektif $f(x)$ dapat dikalkulasi sebagai jumlah angka-angka yang terasosiasi dengan warna yang digunakan untuk mewarnai X.

$$f(X) = \sum_{k=1}^{n} k \cdot pgt\left(\sum_{l=1}^{n} x_l^k \right)$$

Konstrain $Gj(X)$ menjamin tidak adanya sisi-sisi yang memiliki simpul-simpul berwarna sama yang dihubungkan olehnya.

$$G_j(X) = \sum_{[v_i, v_j] \in E} x_l^{jk} x_k^j \le 0 \qquad 1 \le j \le n$$

Pada [2], ditunjukkan bahwa Algoritma Semut mempu mengungguli metode-metode klasik untuk menyelesaikan Permasalahan Pewarnaan Graf seperti *Simulated Annealing* dan Algoritma Genetik.

Selain Metode yang sudah dijelaskan sebelumnya, terdapat pula berbagai pengembangan lain dari pemanfaatan Algoritma Semut untuk menyelesaikan Permasalahan Pewarnaan Graf.

Salah satu dari pengembangan lain tersebut yang cukup dikenal dinamakan *AS-ATP*. *AS-ATP* menggunakan pendekatan yang sama dengan Algoritma Semut yang sudah dijelaskan. Namun, terdapat pengembangan utama, yaitu setiap "semut" harus membuat dua buah pilihan dan meninggalkan 2 buah jejak *Pheromone.* Karena algoritma ini berlaku juga tidak hanya untuk masalah pewarnaan graf, maka akan dijelaskan dalam terminologi globalnya yakni: "Semut-semut" memilih suatu *item* tertentu dan baru menentukan ke *resource* mana *item* tersebut akan ditetapkan. 2 buah fungsi aturan probabilistik dari 2 buah jejak *Pheromone* τ_1 dan τ_2 serta 2 buah nilai heuristik η_1 dan η_2 digunakan untuk menentukan 2 buah pilihan tersebut. Setiap "semut" pada koloni membuat solusi yang memungkinkan dalam setiap siklusnya. Pada tiap akhir siklus, jejak *Pheromone* diperbaharui berdasarkan kepada kualitas dari solusi-solusi yang dibentuk oleh "semut-semut" tersebut. Solusi terbaik yang ditemukan disimpan oleh algoritma. Langkah-langkah tersebut diulang berkali-kali hingga menemukan solusi yang merupakan solusi terbaik dari permasalahan yang ada.

Terdapat pula variasi lain yang dikenal dengan nama Algoritma ANTCOL untuk penyelesaian masalah pewarnaan graf. Algoritma tersebut menggunakan heuristik pewarnaan graf yang sudah diketahui untuk meningkatkan performansi dari algoritma tersebut. Hasil tes performansi algoritma untuk pewarnaan graf menunjukkan bahwa algoritma ANTCOL juga memiliki performansi hampir sama baiknya dengan 2 jenis variasi Algoritma Semut yang telah dijelaskan sebelumnya.

4. Kesimpulan dan Saran

Algoritma Semut dan berbagai pengembangan dari Algoritma Semut merupakan suatu topik menarik untuk dipelajari sebab mengambil inspirasi berdasarkan tingkah laku semut sebenarnya. Berbagai pengembangan yang dilakukan untuk mengatasi permasalah pewarnaan graf menunjukkan hasil optimasi yang cukup baik dibandingkan dengan algoritma-algoritma klasik untuk menyelesaikan permasalahan pewarnaan graf.

Tiga buah variasi Algoritma Semut yang telah dijelaskan di atas memiliki konsep dasar yang sama dan memiliki performansi sama baiknya dalam menangani kasus pewarnaan graf. Pengembangan Algoritma Semut di bidang lainnya pun sampai saat ini masih merupakan topik menarik yang selalu berusaha dikembangkan oleh komunitas ilmiah di seluruh dunia.

Referensi

[1] Munir,Rinaldi.2004.Diktat Kuliah Matematika Diskrit. Departemen Teknik Informatika ITB

[2] Comellas F. and Oz_on J., Graph colouring algorithms for assignment problems in radio networks, Applications of Neural Networks to Telecommunications 2. Edit. J.Alspector, R. Goodman y T.X.Brown, Lawrence Erlbaum Ass., pp. 49-56, 1995. http://www-mat.upc.es/~comellas/radio/radio.html

[3] Costa D. and Hertz A., Ants can colour graphs, Journal of the Operational Research Society (1997) 48, 295-305

[4] M. Dorigo, V. Maniezzo and A. Colorni: *The Ant System: Optimization by a Colony of Cooperating Agents*, IEEE Transactions on Systems, Man, and Cybernetics-Part B, No. 26(1), 1996, pp. 29-41.