Studi Pohon Steiner dan Penggunaannya dalam Perancangan *Chip* dan Jaringan

Samuel Simon – NIM: 13506032
Program Studi Teknik Informatika ITB, Bandung
Email: if16032@students.if.itb.ac.id

Abstrak – Makalah ini membahas tentang salah satu pengembangan dari teori pohon yang dikenal dengan nama pohon Steiner, serta penggunaan pohon Steiner dalam perancangan *chip* dan desain jaringan. Pemberian nama pohon Steiner dilakukan untuk menghormati seorang matematikawan Swiss yang bernama Jakob Steiner. Teori pohon Steiner serupa dengan teori pohon merentang minimum, namun terdapat beberapa hal khusus yang ditambahkan dalam teori tersebut.

Pada awalnya, teori pohon Steiner dikembangkan untuk membuat desain suatu jaringan sehingga diperoleh jalur terpendek yang dapat menghubungkan seluruh tempat (*node*), tanpa membuat suatu sirkuit. Namun, pada pengembangan selanjutnya, teori ini banyak digunakan pada bidang-bidang lain, khususnya pada bidang perancangan jalur, seperti perancangan jalur pada *chip* (keping) elektronika.

Dalam makalah ini pula, akan dibahas mengenai algoritma pembentukan pohon Steiner disertai dengan analisis dan pembahasan mengenai kompleksitas dari algoritma tersebut. Pada bagian akhir makalah ini, akan dibahas juga beberapa hal yang berhubungan dengan optimasi dari algoritma pembentukan pohon Steiner.

Kata Kunci: steiner tree, minimum spanning tree, pohon, graf

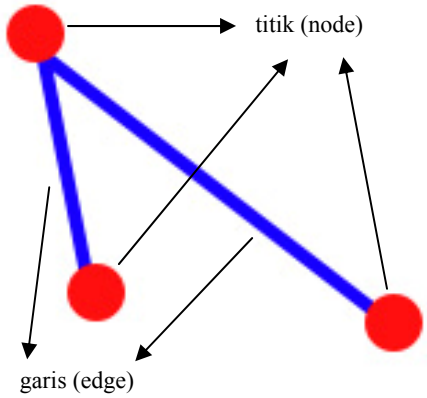
1. PENDAHULUAN

Di dalam dunia matematika dan ilmu komputer, teori graf adalah pokok bahasan yang mempelajari tentang graf, suatu struktur matematis yang biasa digunakan untuk memodelkan sekumpulan objek dan relasi/hubungan di antara objek-objek tersebut. Representasi visual dari graf adalah dengan menggambarkan objek sebagai suatu titik (*vertex/node*) yang biasa dilambangkan dengan huruf V, sedangkan hubungan antar objek digambarkan dengan garis (*edge*) yang biasa dilambangkan dengan huruf E. Sebuah graf dilambangkan dengan G dan didefinisikan sebagai pasangan dari himpunan (V, E). Dalam definisi tersebut:

V : himpunan tidak kosong dari titik-titik (*nodes*) = $\{v_1, v_2, v_3, .., v_n\}$

E : himpunan sisi (edges) yang menghubungkan sepasang titik = $\{e_1, e_2, e_3, .., e_n\}$,

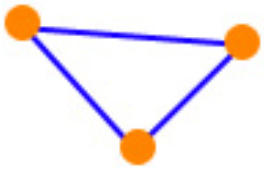
sehingga sebuah graf dapat ditulis singkat dengan notasi: G = (V,E)



Sebuah graf dapat berupa graf berarah (*directed graph*) atau graf tidak berarah (*undirected graph*), tergantung dari hubungan antar objek-objek di dalam graf tersebut. Jika hubungan antara titik-titik dalam graf bersifat sama untuk kedua arah (hubungan antara titik A dan B sama dengan hubungan antara titik B dan A), maka graf tersebut dikatakan graf tidak berarah. Sedangkan jika hubungan antara titik-titik tersebut berbeda, maka graf tersebut dikatakan graf berarah. Di dalam teori graf, dikenal juga istilah graf terhubung dan sirkuit. Graf terhubung adalah graf yang setiap titiknya dapat dicapai oleh semua titik lain dalam graf tersebut, sedangkan yang dimaksud dengan sirkuit adalah lintasan/jalur yang berawal dan berakhir pada titik yang sama.

Graf terhubung.

Graf tidak terhubung.

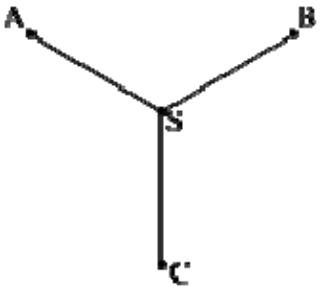


Sirkuit

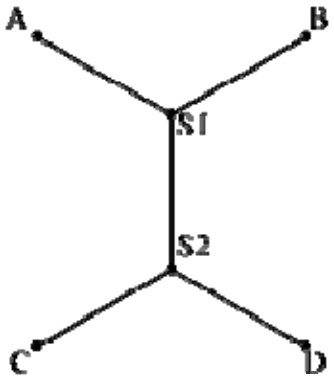
Pengembangan dari teori graf menghasilkan teori yang disebut dengan teori pohon. Pohon adalah suatu graf terhubung yang tidak memiliki arah dan tidak mengandung sirkuit. Dalam teori pohon ini, dikenal istilah pohon merentang (spanning tree). Jika terdapat suatu graf terhubung tak berarah yang bukan pohon (berarti graf tersebut memiliki sirkuit), maka graf tersebut dapat diubah menjadi pohon dengan cara memutuskan sirkuit-sirkuit yang ada. Caranya, mula-mula dipilih sebuah sirkuit, kemudian hapus salah satu garis/sisi dari sirkuit ini. Graf tersebut akan tetap terhubung, namun jumlah sirkuitnya berkurang satu. Jika proses penghapusan sirkuit tersebut dilakukan berulang-ulang hingga tidak ada lagi sirkuit yang tersisa, maka graf tersebut akan menjadi sebuah pohon, yang dinamakan pohon merentang. Pohon Steiner serupa dengan pohon merentang, namun pohon Steiner memiliki beberapa aturan tambahan.

2. POHON STEINER

Pohon Steiner serupa dengan pohon merentang, tetapi ada suatu perbedaan mendasar antara pohon Steiner dengan pohon merentang. Pada pohon Steiner, titik dan garis dapat ditambahkan pada graf untuk mengurangi panjang sisi dari pohon merentang. Titik-titik yang ditambahkan untuk mengurangi panjang sisi pada graf tersebut disebut titik Steiner. Untuk setiap kumpulan titik yang sama, dapat diperoleh pohon Steiner yang berbeda.



Pohon Steiner untuk 3 titik (A, B, C).
S = titik Steiner.



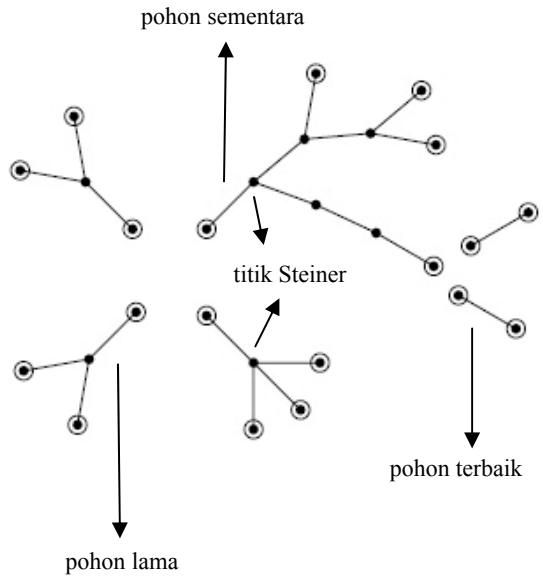
Pohon Steiner untuk 4 titik (A, B, C, D).
S1, S2 = titik Steiner.

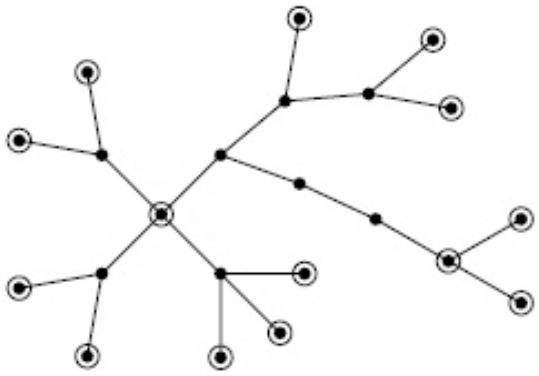
3. ALGORITMA PEMBENTUKAN POHON STEINER

Terdapat beberapa algoritma untuk membentuk pohon Steiner dari suatu graf, salah satunya adalah sebagai berikut:

- Cari titik tengah dari seluruh titik yang akan dihubungkan.
- Urutkan semua titik tersebut berdasarkan jaraknya terhadap titik tengah yang telah dicari sebelumnya.
- Letakkan titik Steiner antara 3 titik pertama, kemudian sambungkan titik Steiner tersebut dengan ketiga titik yang ada di sekitarnya.
- Cari posisi titik Steiner terbaik sehingga didapat pohon Steiner paling optimal untuk ketiga titik pertama tersebut. Ini menjadi pohon Steiner pertama dalam graf tersebut, yang dapat disebut pohon sementara.
- Lakukan pengulangan dari k = 4, … , n untuk langkah-langkah:
  - Simpan pohon sementara sebagai pohon lama, dan buat variabel penampung pohon terbaik dengan panjang ∞.
  - Untuk setiap sisi (a; b) dari pohon sementara, lakukan:
    - Letakkan titik Steiner (s) pada sisi (a; b)
    - Hapus sisi (a; b)
    - Tambahkan sisi (t_k; s), (a; s), (b; s)
    - Cari posisi s terbaik sehingga didapat pohon Steiner yang paling optimal untuk titik tersebut.
    - Jika hasil pohon tersebut memiliki panjang yang lebih kecil daripada pohon terbaik, simpan pohon ini menjadi pohon terbaik.
    - Ubah pohon sementara menjadi pohon lama.
  - Simban pohon sementara menjadi pohon terbaik.

- Simpan pohon hasil proses tersebut menjadi pohon Steiner akhir.





Hasil akhir pohon Steiner.

Algoritma lain yang banyak untuk membuat pohon Steiner dikenal dengan nama Dreyfus-Wagner. Algoritma ini akan penulis bahas dalam perhitungan kompleksitas algoritma pembentukan pohon Steiner.

4. KOMPLEKSITAS ALGORITMA PEMBENTUKAN POHON STEINER

Pohon Steiner adalah salah satu kasus NP (non-polinomial) yang cukup terkenal.
Diberikan suatu graf G = (V,E).
Dengan $n = |V|$, sisi dengan panjang $c : E \rightarrow \Re_+$ dan himpunan $Y \subseteq V$ dari $k = |Y|$ titik akhir (*terminal*), akan dicari panjang minimum pohon $T \subseteq E$, yang menghubungkan seluruh Y titik akhir.

Dari sini dan analisis selanjutnya, dikenal sebuah upa-pohon $T \subseteq E$ dari graf G dengan himpunan titik-titiknya. Titik dalam himpunan tersebut dinotasikan dengan $V(T)$. Jadi, sebuah solusi pohon Steiner yang optimal untuk Y adalah pohon $T = T(Y)$ yang memiliki nilai $c(T)$ terkecil, mengacu pada $Y \subseteq V$.

Untuk menghitung kompleksitas, algoritma yang dipakai sebagai bahan analisis adalah algoritma Dreyfus-Wagner. Pertama, perlu dicatat bahwa setiap titik terluar (*leaf*) dari graf harus menjadi titik akhir (*terminal*) dari graf tersebut. Setiap titik di bagian dalam (*interior node*), akan menjadi titik akhir atau titik Steiner. Untuk menjabarkan algoritma Dreyfus-Wagner, diambil notasi berikut: Untuk $X \subseteq V$, $T(X)$ akan menghasilkan panjang minimum dari pohon Steiner untuk X dan pohon Steiner itu sendiri.

Algoritma Dreyfus-Wagner melakukan perhitungan $T(X \cup v)$ secara rekursif untuk setiap $X \subseteq Y$ dan $v \subseteq V$. Pada kasus umum, titik akhir baru $v \subseteq V$ adalah titik terluar dari pohon Steiner $T = T(X \cup v)$ dan v digabungkan oleh jalur terpendek P_{vw} ke titik di bagian dalam $(w \in V(T))$, dengan w memiliki hubungan paling sedikit dengan 3 titik lainnya. Titik w membagi $T\backslash(P_{vw})$ menjadi 2 bagian, yang dinamakan $T(X' \cup w)$ $T(X'' \cup w)$ untuk partisi $X = X' \cup X''$. Oleh karena itu, dapat ditulis:

$$T(X \cup v) = \min(P_{vw} \cup T(X' \cup w) \cup T(X'' \cup w)),$$

dimana nilai minimum diambil dari seluruh partisi $X = X' \cup X''$ dan seluruh $w \in V$.

Proses rekursi di atas juga berlaku untuk "kasus-kasus tidak biasa", dimana titik akhir baru v bukan merupakan titik terluar dari T (dipilih $w = v$) atau ketika v digabung oleh P_{vw} ke titik terluar dari $T(X)$, misalkan saat w hanya terhubung ke 2 titik dalam T.

Dari analisa tersebut, dapat disimpulkan bahwa proses rekursi ini menghitung pohon Steiner optimal dengan tepat untuk $Y \subseteq V$. Pada saat proses tersebut dijalankan, dapat diamati bahwa terdapat kurang dari $n\binom{k}{i}$ himpunan dari $X \cup v$ dengan $|X| = I$ dan setiap X memiliki kurang dari 2^i partisi. Dengan demikian, didapatkan:

$$n \sum_{i \le k} \binom{k}{i} 2^i = n\, 3^k = O^*\left(3^k\right)$$

sebagai batas atas dari waktu komputasi yang diperlukan algoritma tersebut.

5. PENERAPAN POHON STEINER

5.1. Perancangan Desain Jaringan

Desain jaringan digunakan dalam banyak hal, bukan hanya jaringan komputer, namun juga jaringan-jaringan lain, seperti jaringan komunikasi, jaringan

pipa air minum, jaringan jalan, jaringan kabel listrik, dan lain sebagainya. Desain jaringan sangat penting untuk dibuat terlebih dahulu sebelum membuat jaringan aslinya agar segala hal yang berhubungan dengan pembangunan jaringan tersebut dapat direncanakan terlebih dahulu.

Pohon Steiner dapat digunakan untuk merancang jalur-jalur jaringan tersebut. Tempat-tempat yang ingin dilalui oleh jalur jaringan direpresentasikan sebagai titik-titik dalam graf. Kemudian, titik-titik tersebut diproses dengan algoritma yang ada, sehingga didapat pohon Steiner dari titik-titik tersebut. Pohon Steiner tersebut dapat digunakan untuk perancangan jaringan, karena menghasilkan jalur terpendek yang melalui ke seluruh tempat (titik) dalam graf, sehingga dapat membuat proses pembangunan jaringan menjadi lebih efisien.

5.2. Perancangan Chip Elektronika

Pembuatan *chip* elektronika melalui beberapa tahap, diantaranya adalah proses perancangan *chip* itu sendiri. Dalam proses perancangannya, produsen *chip* menetapkan titik-titik yang akan menjadi tempat diletakkannya komponen-komponen elektonika. Setelah semua titik ditentukan, maka tahap selanjutnya adalah perancangan jalur-jalur yang akan menghubungkan titik-titik yang telah dibuat sebelumnya. Jalur-jalur yang dipilih adalah jalur terpendek sehingga proses pembuatan menjadi lebih efisien baik dari segi biaya maupun waktu pembuatan.

Untuk mempermudah pembuatan jalur tersebut, dapat digunakan teori pohon Steiner. Namun, seringkali jalur yang ingin dibuat dibatasi menjadi jalur horizontal dan vertikal saja. Jika pada proses pembuatan terdapat pembatasan seperti hal tersebut, maka dapat digunakan *rectilinear* Steiner.

6. HAMBATAN DALAM PENGAPLIKASIAN POHON STEINER

Permasalahan yang ditemukan dalam teori pohon Steiner adalah untuk mencari pohon Steiner dari suatu graf, digunakan algoritma yang memiliki kompleksitas cukup besar untuk jumlah titik masukan yang besar (> 30). Selain itu, ditemukan beberapa hal lain yang menjadi kendala dalam mencari dan mengaplikasikan pohon Steiner dalam kehidupan sehari-hari.

Beberapa hal yang menjadi hambatan dalam mengaplikasikan pohon Steiner:

- Banyaknya titik yang akan dihubungkan dengan pohon Steiner.
  Pohon Steiner antara 2 titik dapat dicari dengan mudah, sama seperti mencari jalur terpendek antara 2 titik tersebut, sedangkan pohon Steiner untuk setiap titik pada graf G, sama dengan mencari pohon merentang minimum pada graf tersebut. Namun, algoritma pencarian pohon Steiner termasuk kategori NP (*non-polynomial*) *complete*. Dengan demikian, pertambahan banyak titik yang akan dihubungkan menjadikan komplesitas algoritma semakin besar dan tidak dapat ditentukan besaran pertambahannya.

- Adanya aturan penggunaan garis/sisi.
  Pada sebagian besar permasalahan dalam kehidupan nyata, pembentukan pohon Steiner mendapat aturan tambahan, yaitu garis/sisi yang boleh ditambahkan hanya berupa garis horizontal/vertikal saja, seperti pada pembuatan desain *chip*. Permasalahan ini dikenal dengan istilah *rectilinear Steiner*. Permasalahan ini dapat menambah kompleksitas dari algoritma yang digunakan.

- Adanya titik Steiner yang harus dibuat.
  Dalam hampir semua kasus pembuatan pohon Steiner, diperlukan adanya titik Steiner untuk menghasilkan jalur yang terpendek. Namun, permasalahan yang dihadapi adalah titik-titik Steiner tersebut tidak diketahui berapa banyak dan dimana harus diletakan. Hal tersebut menjadi salah satu hal yang menyebabkan kompleksitas algoritma tersebut meningkat.

7. OPTIMASI

Walaupun banyak terdapat kendala dalam penerapan teori pohon Steiner ini, namun terdapat beberapa cara untuk mengoptimalkan pencarian pohon Steiner dari suatu graf. Caranya, buat graf yang menggambarkan titik-titik masukan, dengan panjang sisi (i, j) sama dengan jarak antara titik i dan j. Kemudian cari pohon merentang minimum dari graf ini. Maka akan ditemukan hasil yang merupakan pendekatan baik untuk pohon Steiner biasa maupun *rectilinear* Steiner.

Kasus terburuk untuk cara pendekatan seperti ini adalah jika 3 titik membentuk segitiga sama sisi. Pohon merentang minimun akan mengandung 2 sisi dengan panjang = 2, namun seharusnya panjang pohon Steiner terkecil yang dapat dicapai dengan menggunakan bantuan titik Steiner di tengah-tengah ketiga titik tersebut adalah $\sqrt{3}$. Perbandingan perbedaan hasil ini dan hasil pendekatan adalah $\frac{\sqrt{3}}{2} \approx 0{,}866$. Angka perbandingan perbedaan hasil

ini selalu diperoleh untuk setiap hasil pendekatan dan hasil optimal pada pohon Steiner. Pada *rectilinear* Steiner, perbandingan perbedaan hasil pendekatan dengan hasil optimal selalu $\geq \frac{2}{3} \approx 0{,}667$.

8. KESIMPULAN

Teori pohon, yang merupakan pengembangan lanjutan dari teori graf, banyak diaplikasikan dalam kehidupan sehari-hari, mulai dari bidang komputer, biologi, studi perencanaan, dan lainnya. Salah satu teori yang cukup banyak digunakan adalah teori pohon Steiner. Teori ini digunakan untuk mencari jalur terpendek yang dapat menghubungkan titik-titik dalam suatu graf.

Teori pohon Steiner banyak digunakan pada perancangan *chip* dan desain jaringan. Namun, sayangnya proses komputasi teori ini mempunyai kompleksitas yang cukup besar dan tidak bisa diprediksi untuk jumlah titik yang besar. Hal ini disebabkan karena teori pohon Steiner termasuk permasalahan NP (*non-polynomial*). Analisis pada salah satu algoritma untuk mencari pohon Steiner memperlihatkan bahwa algoritma tersebut memiliki kompleksitas: $O^*\left(3^k\right)$.

Dengan kompleksitas yang cukup besar tersebut, pohon Steiner jarang digunakan untuk titik-titik yang berjumlah banyak (> 50). Hal ini membuat beberapa ilmuwan mencari cara untuk mengurangi waktu proses komputasi. Salah satu cara yang digunakan adalah cara pendekatan. Dengan cara ini, dapat diperoleh hasil pohon Steiner dengan waktu komputasi yang jauh lebih cepat. Namun, cara ini menghasilkan panjang pohon Steiner yang lebih besar dari pada pencarian dengan perhitungan sesungguhnya. Namun, rasio perbedaan itu, cukup kecil sekitar 0,866 untuk pohon Steiner pada kasus biasa.

Terdapat satu jenis pohon Steiner khusus, yang biasa digunakan untuk kasus-kasus tertentu, yaitu *rectilinear* Steiner. *Rectilinear* Steiner adalah pohon Steiner yang hanya memiliki 2 macam garis/sisi saja, yaitu horizontal/vertikal. Jika cara pendekatan digunakan untuk kasus ini, maka rasio perbedaan hasil pohon Steiner yang didapat dengan pohon Steiner optimal menjadi $\geq \frac{2}{3} \approx 0{,}667$.

DAFTAR REFERENSI

[1] B. Aronov, M. Bern, dan D. Eppstein, "On the number of minimal 1-Steiner trees", Discrete and Computational Geometry", 1994, hal. 29-34.

[2] P. Berman dan V. Ramaiyer, "Improved approximations for the Steiner tree problem", In Proceedings of the Third Symposium on Discrete Algorithms, 1992, hal. 325-334.

[3] P. Berman dan V. Ramaiyer, "Improved approximations for the Steiner tree problem", Journal of Algorithms, no. 17, hal. 381-408.

[4] E. J. Cockayne dan D. E. Hewgill, "Improved Computation of Plane Steiner Minimal Trees", Algorithmica, no. 7, 1992, hal. 219-229.

[5] Steiner tree - Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Steiner_tree Tanggal akses: 27 Desember 2007 pukul 17.30 GMT +7

[6] Graph - Wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/Graph Tanggal akses: 29 Desember 2007 pukul 08.00 GMT +7

[7] Steiner tree http://www.nist.gov/dads/HTML/steinertree.html Tanggal akses: 29 Desember 2007 pukul 08.00 GMT +7