PERBANDINGAN ALGORTIMA PRIM DAN KRUSKAL DALAM MENENTUKAN POHON RENTANG MINIMUM

Kodirun[1]
[1]Jurusan Matematika FMIPA Universitas Haluoleo, Kendari
e-mail: kodirun_zuhry@yahoo.com

Abstrak

Masalah yang sering ditemukan di dalam graf adalah bagaimana menentukan jarak minimal atau jarak terpendek, misalnya dalam bidang transportasi, pemasangan kabel listrik, dan kabel telepon. Salah satu cara untuk menyelesaikan masalah tersebut yaitu dengan penentuan pohon rentang minimum. Penelitian ini membandingan dua algoritma yaitu prim dan kruskal dalam menentukan rentang minimum suatu pohon. Hasil yang diperoleh menunjukkan bahwa algoritma prim lebih efisien dibanding algoritma kruskal saat graf yang diberikan memiliki banyak sisi dengan simpul yang sedikit (graf lengkap), tetapi algoritma kruskal lebih efisien dibanding algoritma prim saat graf yang diberikan memiliki banyak simpul dengan sisi yang sedikit.

Kata-kata kunci: *graf, lintasan terpendek, algoritma prim, algoritma kriskal*.

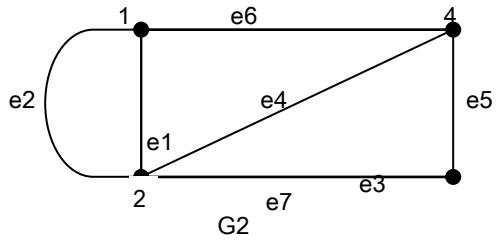
I. LATAR BELAKANG

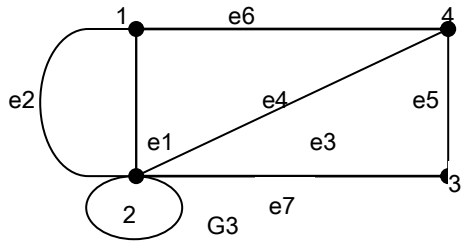
Graf adalah pasangan himpunan (V, E) dengan V adalah himpunan tidak kosong dari simpul-simpul (vertices) dan E adalah himpunan sisi-sisi (edges) yang menghubungkan sepasang simpul pada graf tersebut. Graf digunakan untuk memodelkan suatu masalah sehingga menjadi lebih mudah, yaitu dengan cara merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut.

Masalah yang sering ditemukan di dalam graf adalah bagaimana menentukan jarak minimal atau jarak terpendek, misalnya dalam bidang transportasi, pemasangan kabel listrik, dan kabel telepon. Salah satu cara untuk menyelesaikan masalah tersebut yaitu dengan penentuan pohon rentang minimum. Banyak algoritma dalam menentukan jalur terpendek suatu graf, namun belum ada satupun yang bisa diklaim sebagai yang terbaik. Oleh karena itu tujuan penelitian ini membandingan dua algoritma yaitu prim dan kruskal dalam menentukan rentang minimum suatu pohon.

II. TINJAUAN PUSTAKA

Graf G (V,E) adalah suatu himpunan berhingga tak kosong dari objek-objek yang disebut simpul (minimal tunggal) bersama-sama dengan suatu himpunan yang anggotanya adalah pasangan yang tak berurut dari simpul-simpul yang berbeda disebut sisi (mungkin kosong). Himpunan simpul dari G dinotasikan dengan V dan himpunan sisi dinotasikan dengan E. Gambar 1 merupan contoh graf tak sederhana.

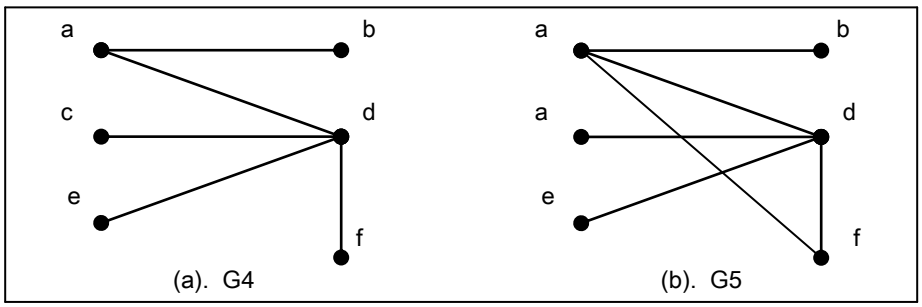




Gambar 1. Contoh graf tak sederhana G2 dan G3

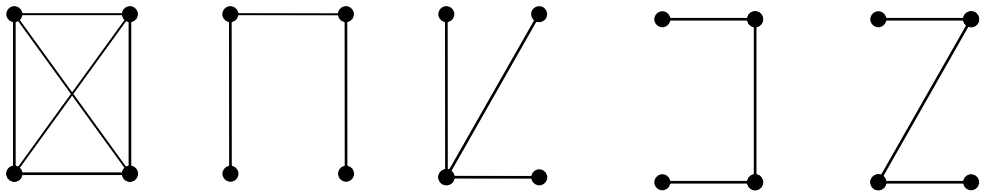
Gelang (Loop) adalah sisi yang menghubungkan sebuah simpul dengan dirinya sendiri. Jika terdapat lebih dari satu sisi yang menghubungkan dua simpul, maka sisi-sisi tersebut dinamakan sisi ganda. Pada G2 sisi e1 =(1,2) dan sisi e2 =(1,2) dinamakan sisi ganda karena kedua sisi ini menghubungkan dua simpul yang sama yaitu simpul 1 dan simpul 2. Sedangkan pada graf G3 sisi e7 =(2,2) dinamakan gelang karena berawal dan berakhir di simpul yang sama.

Beberapa terminologi yang sering digunakan dalam mempelajari graf yaitu: (1) lintasan (path), barisan berhingga dari simpul dan sisi suatu graf, berganti-ganti yang mulai dan berakhir dengan simpul, dengan syarat bahwa setiap sisi dalam barisan itu adalah sisi yang bersamaan dengan simpul yang langsung mendahului dan mengikuti sisi itu dalan barisan tersebut (Suryanto,1986); (2) sirkuit (cycle), lintasan dengan semua simpul yang dilalui hanya muncul satu kali dengan simpul pertama sama dengan simpul terakhir (Wilson & Watkins,1982); (3) bertetangga (adjacent) dan bersisian (incidency), dua buah simpul pada graf dikatakan bertetangga (adjacent) bila keduanya berhubungan langsung dengan sebuah sisi dan suatu sisi e dikatakan bersisian (icidency) dengan simpul vi dan vj, jika e=(vi,vj) atau sisi e menghubungkan simpul vi dan vj. Pohon adalah graf tak berarah terhubung yang tidak mempunyai sirkuit (Munir,2005). Misalkan G4 (V,E) pada Gambar 2 adalah graf tak berarah sederhana dan banyak simpulnya n, maka graf dikatakan sebuah pohon apabila memiliki sifat-sifat : (1) setiap pasangan simpul di dalam G4 terhubung dengan lintasan tunggal, (2) G4 terhubung dengan memiliki jumlah sisi m = n-1 sisi, (3) G4 tidak mengandung sirkuit dan memiliki jumlah sisi m = n-1 sisi, (4) G4 tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membangun hanya satu sirkuit, dan (5) G4 terhubung dengan semua sisinya adalah jembatan (jembatan adalah sisi bila dihapus menyebabkan graf terpecah menjadi dua komponen).



Gambar 2. (a) G4 merupakan pohon, (b) G5 bukan pohon

Misalkan G (V,E) adalah graf tak-berarah terhubung yang bukan pohon, yang berarti di G terdapat beberapa sirkuit. G dapat diubah menjadi pohon, yang berarti di G terdapat beberapa sirkuit. G dapat diubah menjadi pohon T = (V1,E1) dengan cara memutuskan sirkuit-sirkuit yang ada. Mula-mula dipilih sebuah sirkuit, lalu dihapus satu sisi dari sirkuit ini. G akan tetap terhubung dan jumlah sirkuitnya berkurang satu. Bila ini dilakukan berulang-ulang sampai semua sirkuit di G hilang, maka G menjadi satu pohon T, yang dinamakan pohon merentang (spanning tree). Dikatakan pohon merentang jika semua simpul pada pohon T sama dengan semua simpul pada graf G dan sisi-sisi pada pohon T $\subseteq$ sisi-sisi pada graf G . Dengan kata lain V1 =V dan E1 $\subseteq$ E (Munir, 2005) (lihat Gmbar 3).



Gambar 3. Graf lengkap G6 dengan empat pohon merentangnya : T1, T2, T3, dan T4,

Tiap-tiap sisi yang diusulkan memiliki suatu beban tak negatif yang berkaitan dengannya (Bronson & Wosparkrik, 1988). Jika G adalah graf berbeban, maka beban merentang T dari G didefenisikan sebagai jumlah beban semua di T pohon merentang yang berbeda mempunyai beban yang berbeda pula. Di antara semua pohon merentang di G pohon merentang yang berbeban minimum dinamakan pohon merentang minimum (minimum spanning tree) (Munir, 2005).

Algoritma prim:

1. Ambil sisi graf G yang berbeban minimum, masukan ke dalam T.
2. Pilih sisi e yang mempunyai beban minimum dan bersisian dengan simpul di T, tetapi e tidak membentuk sirkuit di T, masukan e ke T.
3. Ulangi langkah dua sebanyak (n–2) kali, (n = banyak titik).

Algoritma kruskal:

1. T masih kosong.
2. Pilih sisi e dengan beban minimum yang tidak membentuk sirkuit di T masukan ke dalam T.
3. Ulangi langkah dua sebanyak (n-1) kali, (n = banyak titik).

III. METODE PENELITIAN

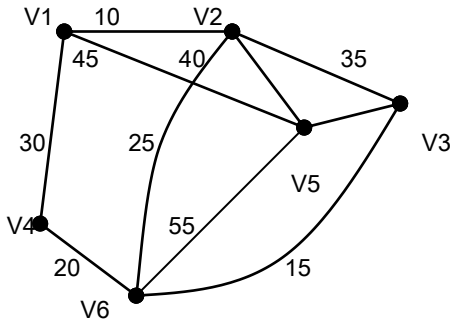
Adapun metode yang akan digunakan dalam menyelesaikan penelitian ini adalah metode pengumpulan data dari instansi yang terkait dan metode kepustakaan (*Library Research*).

IV. HASIL DAN PEMBAHASAN

IV.1 Kasus 1.

Sebuah perusahaan yang bergerak dalam bidang telekomunikasi membangun jaringan baru yang menghubungkan dengan semua wilayah dari jaringan yang lama dengan asumsi semakin

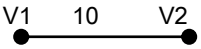
panjang kabel yang dipasang semakin mahal biaya yang harus dikeluarkan. Jaringan dibangun memakan biaya sesedikit mungkin. Dengan bentuk graf sebagai berikut :



Gambar 4. Model Jaringan lama

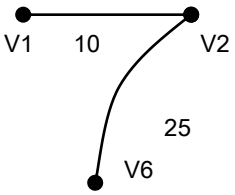
Algoritma Prim

Langkah 1: Mengambil sisi graf G yang berjarak minimum, kemudian dimasukkan ke dalam pohon merentang (T) sisi (V1,V2) dengan jarak 10 km maka pohon merentang yang terbentuk adalah



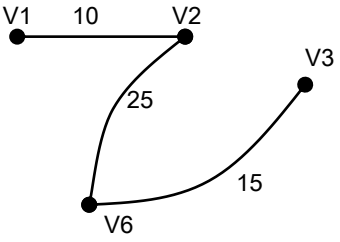
Gambar 5. Pohon Merentang Sisi (V1,V2)

Langkah 2: Memilih e1 yang mempunyai jarak minimum dan bersisian dengan titik di T, tetapi e1 tidak membentuk sisi (V2,V6) dengan jarak 25 (satuan) maka pohon merentang yang dibentuk



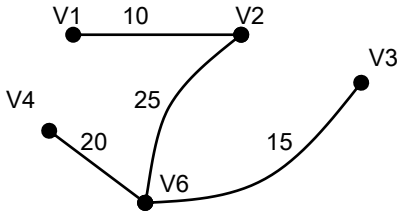
Gambar 6. Pohon Merentang Penambahan Sisi (V1,V2)

Sisi (V3,V6) dengan jarak 15 km maka pohon merentang yang dibentuk adalah



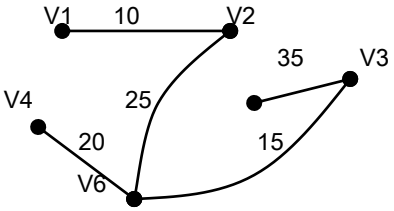
Gambar 7. Pohon Merentang Penambahan Sisi (V3,V6)

Sisi (V4,V6) dengan jarak 20 km maka pohon merentang yang dibentuk adalah



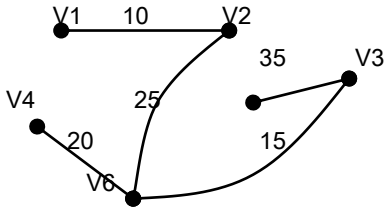
Gambar 8. Pohon Merentang Penambahan Sisi (V4,V6)

Sisi (V3,V5) dengan jarak 35 km maka pohon merentang yang dibentuk adalah



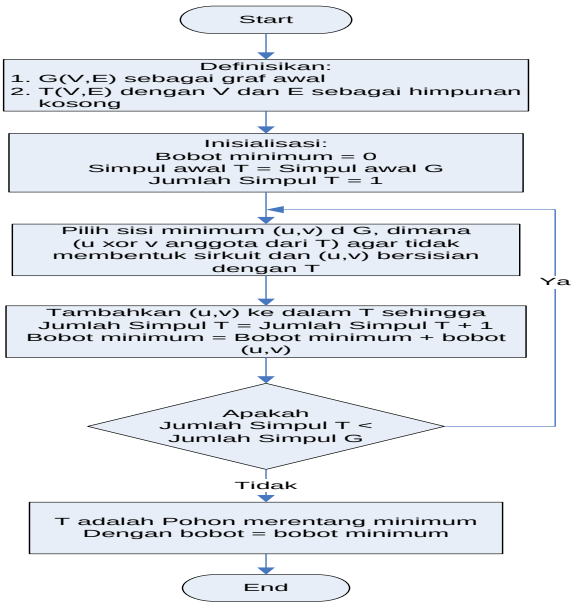
Gambar 9. Pohon Merentang Penambahan Sisi (V3,V5)

Karena semua sisi sudah terhubungkan maka tidak ada lagi sisi yang bisa digunakan, seingga diperoleh jarak minimumnya adalah 10+25+15+20+35=105 km dengan model pada Gambar 10.

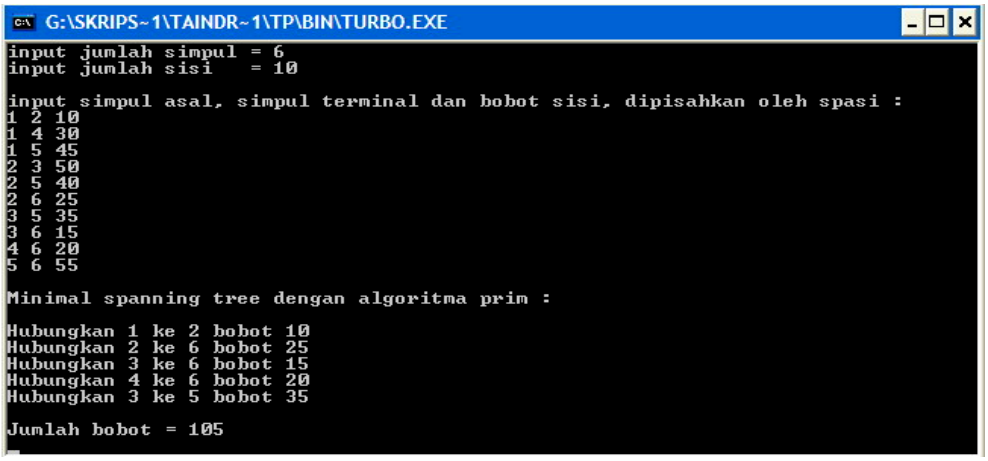


Gambar 10. Model Jaringan Baru

Flowchart:



Output :

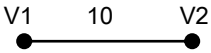


Algoritma Kruskal

Langkah 1 : Pohon merentang masih kosong

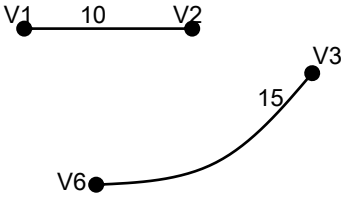
Gambar 11. Pohon Merentang Masih Kosong

Langkah 2: Memilih sisi e1 yang mempunyai jarak minimum yang tidak membentuk sirkuit di T, dimasukkan e1 ke dalam T. Sisi (V1,V2) dengan jarak 10 km maka pohon merentang yang dibentuk adalah



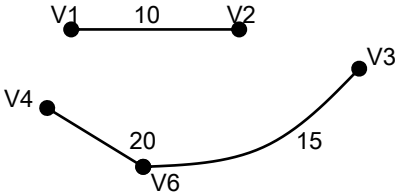
Gambar 12. Pohon Merentang Sisi (V1,V2)

Sisi (V3,V6) dengan jarak 15 km maka pohon merentang yang dibentuk adalah



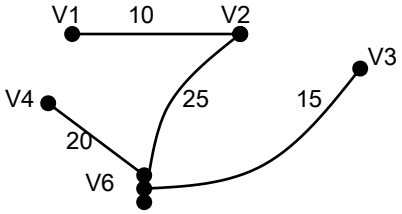
Gambar 13. Pohon Merentang Sisi (V3,V6)

Sisi (V4,V6) dengan jarak 20 km maka pohon merentang yang dibentuk adalah



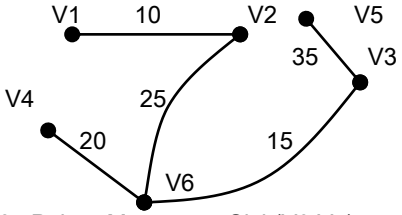
Gambar 14. Pohon Merentang Sisi (V4,V6)

Sisi (V2,V6) dengan jarak 25 km maka pohon merentang yang dibentuk adalah



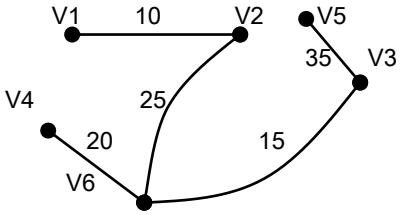
Gambar 15. Pohon Merentang Sisi (V2,V6)

Sisi (V3,V5) dengan jarak 35 km maka pohon merentang yang dibentuk adalah



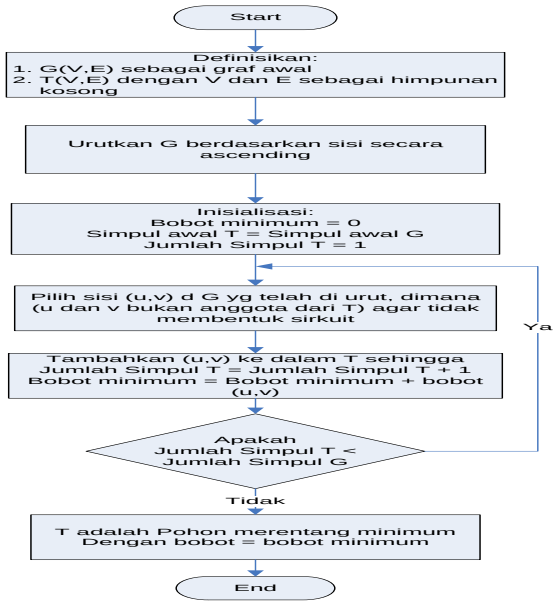
Gambar 16. Pohon Merentang Sisi (V3,V5)

Karena semua jaringan telah terhubung maka pohon merentang yang dihasilkan adalah 10+25+15+20+35=105 km dengan model jalur pada Gambar 17.



Gambar 17. Model Jaringan Baru

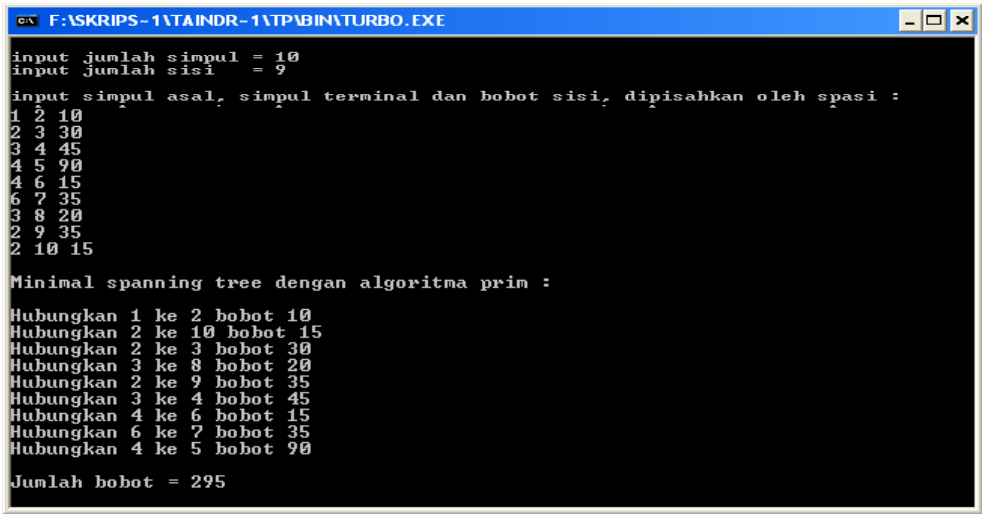
Flowchart :



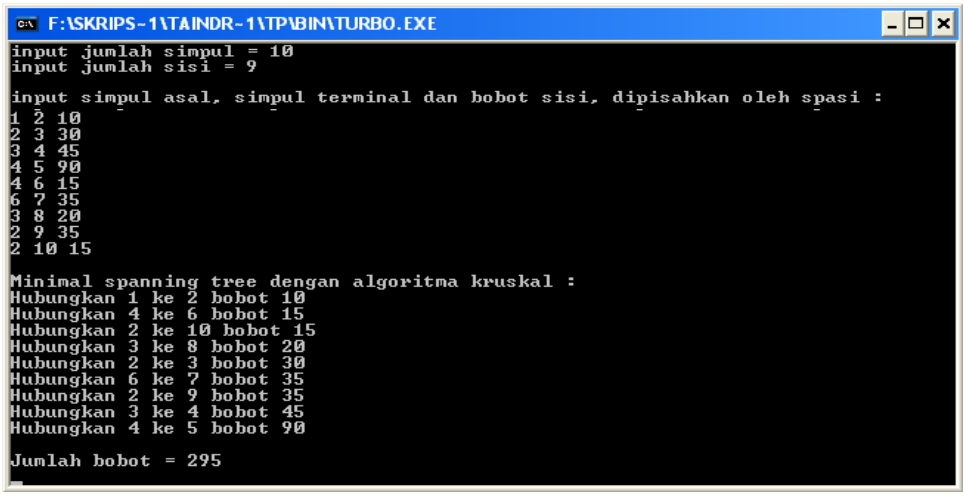
Output:

IV.2. Kasus 2: Banyaknya Simpul Lebih Besar Daripada Banyaknya Sisi

Output Program Algoritma Prim:



Output Program Algoritma Kruskal:



Perbedaan prinsip antara algoritma prim dan algoritma kruskal adalah jika algoritma prim sisi yang dimasukkan ke dalam T harus bersisian dengan sebuah simpul di T, sedangkan algoritma kruskal sisi yang dipilih tidak perlu bersisian dengan sebuah simpul di T asalkan penambahan sisi tersebut tidak membentuk sirkuit. Hal ini mengakibatkan banyaknya jumlah proses perbandingan algoritma prim lebih sedikit dibandingkan dengan algoritma kruskal. Algoritma prim lebih berorientasi kepada pencarian simpul sedangkan algoritma kruskal pada pencarian sisi, dengan sisi-sisi tersebut harus diurutkan dan ini memakan waktu yang cukup lama, apalagi kalau pengurutan menggunakan algoritma standar dengan O(n2) sehingga saat pengujian kasus untuk masukkan yang berbeda algoritma prim unggul saat graf memiliki banyak sisi. Namun saat simpul yang diberikan banyak tetapi sisi yang menghubungkan simpul-simpul itu hanya sedikit, algoritma kruskal bisa lebih unggul.

V. KESIMPULAN

Kesimpulan yang dapat diambil dari studi dan perbandingan dua algoritma pencarian pohon merentang minimum adalah algoritma prim lebih efisien dibanding algoritma kruskal saat graf memiliki banyak sisi dengan simpul yang sedikit (graf lengkap), sebaliknya algoritma kruskal lebih efisien dibanding algoritma prim saat graf memiliki banyak simpul dengan sisi yang sedikit.

VI. DAFTAR PUSTAKA

1. Bronson, R., & Wosparkrik, J.H., 1988. *Teori dan soal-soal Operatian Research*. Erlangga. Jakarta.
2. Munir, R. 2003. *Matematika Diskrit*. Informatika. Bandung.
3. Munir, R. 2005. *Matematika Diskrit*. Informatika. Bandung
4. Wilson J., & Watkins J (terjemahan Theresia M.H Tirta). 1982. *Graf pengantar I*. Universitas Press IKIP, Surabaya.