Menyelesaikan Topological Sort Menggunakan Directed Acyclic Graph

Muhammad Afif Al-hawari (13510020)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
afif.alhawari@s.itb.ac.id

***Abstrak*—Topological sort merupakan salah satu kasus unik yang sering kita temui dalam dunia ini. Dalam topological sort ini, masalah yang dibahas adalah mengenai bagaimana mengurutkan suatu himpunan elemen dengan melihat hubungan dari setiap anggota himpunan tersebut. Beberapa anggota himpunan ada yang harus di letakkan lebih awal dari anggota lainnya. Menggunakan DAG (Directed Acyclic Graph), kasus topological sort ini bisa divisualisasikan dengan sangat baik. Kita bisa mengurutkan dengan mudah elemen-elemen tersebut menggunakan metode ini.**

***Kata Kunci*—Graf, Topological Sort, DAG (Directed Acyclic Graph)**

I. PENDAHULUAN

Kasus Topological Sort merupakan salah satu hal yang menarik yang banyak kita temui aplikasinya dalam dunia ini. Persoalan yang didapatkan adalah bagaimana kita mengurutkan suatu elemen-elemen yang memiliki hubungan keterurutan parsial, menjadi suatu kumpulan elemen yang terurut secara linear.

Kasus pengurutan seperti ini banyak sekali kita temui, tidak hanya dalam dunia informatika saja. Sebagai contoh, ketika kita mengerjakan suatu proyek. Dalam proyek tersebut ada kumpulan-kumpulan pekerjaan yang harus diselesaikan. Ada beberapa pekerjaan yang baru bisa diselesaikan ketika pekerjaan yang lainnya terselesaikan. Sehingga urutan pekerjaan yang harus dilakukan menggunakan prinsip topological sort. Sebagai ilustrasi tambahan, dalam proyek pembuatan bangunan. Dalam proyek ini, untuk membuat dinding terlebih dahulu kita harus membuat pondasi terlebih dahulu. Setelah membuat dinding, barulah kita dapat membuat pintu.

Kasus Topological Sort ini bisa diilustrasikan menggunakan graf. Topological Sort ini mungkin dilakukan jika hubungan antar elemen himpunan tidak membentuk suatu siklus, sehingga tidak bisa ditentukan mana yang harus diletakkan lebih awal. Untuk itu representasinya bisa menggunakan DAG (Directed Acyclic Graph) yaitu sebuat graf berarah yang tidak memiliki sirkuit yang terarah.

Pada makalah ini, kita akan membahas contoh-contoh kasus dari topological sorting, serta penyelesaiannya menggunakan Directed Acyclic Graph.

II. DASAR TEORI

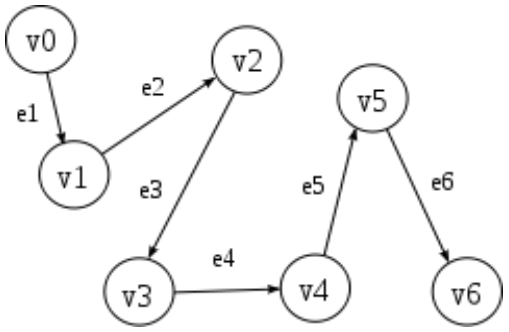
1. Graf

Graf digunakan untuk merepresentasikan objek-objek diskrit. Secara geometri graf digambarkan sebagai kumpulan simpul-simpul (*vertex*) yang dihubungkan dengan kumpulan garis-garis (*edge).* Simpul-simpul ini menggambarkan objek-objek diskrit yang direpresentasikan, sedangkan garis menggambarkan hubungan antara objek-objek diskrit tersebut.

Secara matematis, graf didefinisikan sebagai pasangan himpunan (V,E) dengan :

G = {V,E},

G : Graf
V : Himpunan tidak kosong dari simpul-simpul
E : Himpunan Sisi-Sisi yang menghubungkan simpul-simpul.



Gambar 2.1 Contoh Graf

Graf ini sendiri bisa dibagi menjadi beberapa jenis, tergantung sifatnya. Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, graf digolongkan menjadi dua, yaitu graf sederhana dan graf tak-sederhana.

Graf sederhana adalah graf yang tidak mengandung gelang maupun sisi ganda. Gambar 2.1 merupakan salah satu contoh graf sederhana. Sedangkan graf tak-sederhana adalah graf yang mengandung sisi ganda atau gelang.

Berdasarkan orientasi arah pada sisi, graf dapat

dikelompokkan lagi menjadi dua. Yang pertama adalah graf tak-berarah. Pada graf tak berarah ini, setiap sisinya tidak mempunyai orientasi arah. Yang kedua adalah graf berarah. Pada graf berarah ini, setiap sisi pada graf memiliki orientasi arah tertentu. Sebagai contoh, graf pada gambar 2.1 merupakan graf berarah.

Pada Graf, terdapat beberapa terminologi dasar yang penting. Terminologi tersebut akan mendukung pembahasan pada makalah ini. Terminologi-terminologi itu antara lain :

a. Ketetanggaan
   Dua simpul dikatakan bertetangga apabila keduanya terhubung langsung.
b. Bersisian
   Sembarang sisi $e(v_j,v_k)$ dikatakan bersisian dengan simpul v_j dan v_k
c. Simpul Terpencil
   Simpul yang tidak memiliki sisi yang bersisian dengannya.
d. Graf Kosong
   Graf yang sisinya merupakan himpunan kosong
e. Derajat
   Jumlah sisi yang bersisian dengan suatu simpul
f. Lintasan
   Banyak barisan sisi dari suatu simpul ke simpul lainnya.
g. Siklus atau Sirkuit
   Lintasan yang berawal dan berakhir pada simpul yang sama
h. Terhubung
   Dua buah simpul terhubung jika terdapat lintasan dari satu simpul ke simpul yang lainnya.

2. Topological Sort

Topological sort ini merupakan suatu kasus pengurutan, dimana pengurutan dilakukan terhadap elemen-elemen pada suatu himpunan, dengan mempertimbangkan keterurutan parsial dari elemen-elemen tersebut.

Pengurutan topological ini biasanya memberikan beberapa alternatif jawaban. Alternatif jawaban ini terjadi karena ada elemen-elemen yang saling tidak bergantung sehingga bisa diletakkan lebih awal atau lebih akhir elemen lainnya.

Ada banyak kasus yang dapat diselesaikan dengan menggunakan metode topological sort ini. Contoh Ilustrasi kasus yang sering kita temui antara lain :

a. Dalam suatu kurikulum atau perkuliahan, suatu mata kuliah dapat memiliki prerequisite berupa mata kuliah lain. Sebagai contoh, Mata kuliah Matematika Teknik memiliki prerequisite berupa Mata Kuliah Kalkulus IA. Sehingga untuk mengambil mata kuliah matematika teknik seseorang harus mengambil mata kuliah kalkulus IA terlebih dahulu. Namun untuk kasus lain, mata kuliah Struktur diskrit tidak memiliki prerequisite sehingga bisa diambil bersamaan dengan Mata Kuliah Matematika Teknik ataupun setelah atau sebelum mengambil Mata Kuliah Matematika Teknik, sehingga terdapat beberapa alternatif posisi pengurutan Mata Kuliah Struktur Diskrit.
   Pada kasus ini hasil pengurutan yang didapat berupa urutan mata pelajaran yang bisa diambil secara linear.
b. Salah satu contoh kasus yang lain adalah dalam suatu proyek pembangunan. Pembuatan dinding harus dilakukan setelah pembuatan pondasi selesai. Dan pembuatan pintu dan jendela baru bisa dilakukan setelah pembuatan dinding selesai. Dalam kasus ini pengurutan topologi yang bisa kita dapatkan adalah sebagai berikut :
   Pondasi > Dinding > Pintu > Jendela
   Pondasi > Dinding > Jendela > Pintu
   Terdapat dua buah alternatif pengurutan karena pintu dan jendela sama-sama baru bisa dikerjakan setelah dinding selesai, tetapi pintu dan jendela pekerjaan tidak tergantung satu sama lain.
c. Dalam dunia informatika, sebagai contoh bisa diambil dalam bahasa pemograman. Dalam suatu bahasa pemograman, peletakan suatu prosedur dalam teks program harus dilakukan sedemikian rupa sesuai dengan urutan pemanggilan prosedur.

3. Directed Acyclic Graph (DAG)

Directed Acyclic Graph merupakan sebuah graf berarah yang tidak memiliki siklus yang terarah. Graf ini terbentuk dari kumpulan simpul dan sisi, dimana setiap sisi menyambungkan satu simpul ke simpul lainnya. Pada Setiap simpul, tidak ada lintasan berarah yang menghubungkan antara simpul tersebut dengan dirinya sendiri.

DAG ini paling sering digunakan untuk memodelkan kumpulan objek-objek yang terurut secara sekuensial. Salah satu contohnya adalah dalam kasus topological ordering / topological sort.

Pada topological sort menggunakan DAG, setiap elemen dalam himpunan yang akan diurutkan dinyatakan sebagai suatu simpul. Hubungan keteruturan parsial antar elemen dinyatakan dengan suatu sisi berarah. Suatu simpul yang ditunjuk oleh simpul lainnya menandakan bahwa dalam pengurutan simpul yang ditunjuk harus diletakkan setelah simpul yang menunjuk kepadanya.

Sebagai ilustrasi salah satu contoh kasus yang topological sorting yang dapat direpresentasikan dengan DAG :

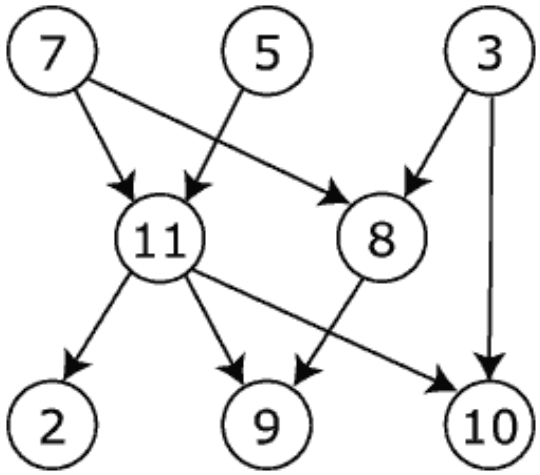
Diberikan himpunan-himpunan keterurutan parsial dari suatu elemen, yaitu :

$3 < 8$, $3<10$, $5<11$, $7 <11$, $7<8$,
$11<2$, $11<9$, $11< 10$, $8<9$

Tanda ‘<’ menyatakan bahwa elemen pertama harus diletakkan lebih dulu daripada elemen kedua.

Jika nilai-nilai tersebut direpresentasikan dengan DAG,

maka didapatkan hasil sebagai berikut :



Gambar 2.2 Representasi dengan Graf

Pada gambar diatas kita bisa melihat bagaimana representasi kasus topological sort dengan suatu Directed Acyclic Graph. Simpul-simpul menunjukkan elemen dan tanda panah menunjukkan hubungan keterurutan parsial, sesuai dengan uraian diatas.

Sebuah Directed Acyclic Graph dapat paling sedikit satu cara pengurutan secara topologi.

4. Algoritma pengurutan

Kembali merujuk pada gambar 2.2 mengenai representasi graf dari kasus topological search sebelumnya. Setelah membuat representasi DAG ini, masalah kita selanjutnya adalah bagaimana mengurutkan elemen ini agar dapat direpresentasikan dalam suatu list linier.

Ide penyelesaian dari masalah ini kira-kira diilustrasikan sebagai berikut :

Misal L adalah sebuah list yang berisi urutan linier dari elemen-elemen pada graf. Sebelum graf diurutkan isi dari list L masih kosong.

Langkah pertama yang harus dilakukan adalah mencari simpul/elemen pada graf gambar 2.2 yang tidak ditunjuk oleh simpul manapun. Pada kasus ini, simpul yang tidak ditunjuk adalah simpul 7, simpul 5 dan simpul 3. Ambil salah satu dari tiga simpul tersebut, kemudian masukkan elemennya ke dalam list L tadi sebagai elemen terakhir. Elemen yang telah dimasukkan kedalam list tadi kita hilangkan dari graf. Kemudian ulangi langkah-langkah tadi pada graf yang telah dihapus tadi elemennya hingga elemen-elemen graf pada gambar 2.2 habis.

Untuk kasus ini, alternatif pengurutan yang didapatkan setelah menggunakan langkah pengurutan tadi adalah :

- 7,5,3,11,8,2,9,10
- 3,5,7,8,11,2,9,10
- 3,7,8,5,11,10,2,9
- 5,7,3,8,11,10,9,2
- 7,5,11,3,10,8,9,2
- 7,5,11,2,3,8,9,10

Alternatif-alternatif jawaban yang berbeda ini didapat karena perbedaan mengambil urutan penghapusan pada graf, jika ada lebih dari satu simpul yang tidak ditunjuk oleh simpul-simpul lainnya.

III. ANALISIS DAN PEMBAHASAN

Pada bagian ini kita akan mencoba menyelesaikan suatu kasus menarik lainnya mengenai topological sort ini. Kita akan menyelesaikannya secara bertahap sesuai dengan langkah-langkah yang telah diuraikan pada bagian II makalah ini.

Diberikan sebuah kasus, dalam suatu proyek, terdapat 8 bagian pekerjaan yang harus diselesaikan. Misal setiap pekerjaan yang harus diselesaikan direpresentasikan dengan menggunakan alfabet A hingga H. Pekerjaan-pekerjaan dalam proyek tersebut saling tergantung satu sama lain. Suatu pekerjaan mungkin saja baru bisa dapat dikerjaan setelah pekerjaan lainnya diselesaikan.

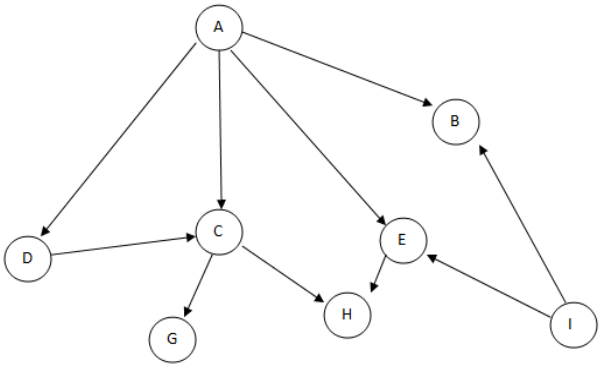
Pekerjaan A merupakan pekerjaan yang harus diselesaikan terlebih dahulu, sebelum pekerjaan D, C, E, dan B bisa dikerjakan. Pekerjaan H harus menunggu pekerjaan C dan E selesai dilakukan. Pekerjaan I merupakan pekerjaan yang dibutuhkan untuk menyelesaikan pekerjaan B dan E. Pekerjaan C dilakukan setelah pekerjaan D selesai, dan merupakan bagian penting untuk menyelesaikan pekerjaan G dan H.

Dalam kasus ini, kita harus menentukan urutan pengerjaan setiap bagian pekerjaan dalam menyelesaikan proyek ini. Dalam menentukan urutan pekerjaannya, kita harus menggunakan prinsip topological sort, serta representasinya menggunakan DAG.

Sebelum membuat representasi dengan DAG, kita perlu mentranslasikan kalimat pernyataan hubungan dari setiap pekerjaan. Menggunakan simbol matematika, kita bisa mendapatkan representasi hubungan keterurutan parsial sebagai berikut :

A<B, A<D, A<C, A<E, C<H,
E<H, I< B, I<E , D<E, C<G,

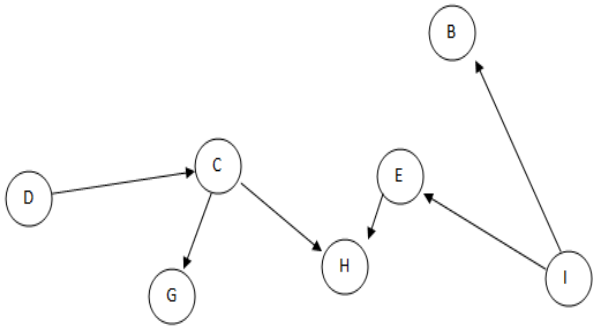
Seperti prosedur pada bagian sebelumnya, hubungan keterurutan parsial ini kita gambarkan sebagai DAG. Hasil yang didapat adalah sebagai berikut :



Gambar 3.1 Representasi proyek dalam DAG

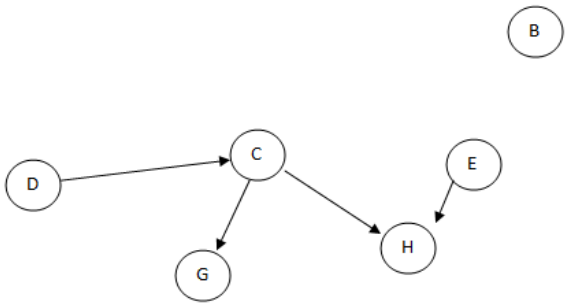
Pada gambar 3.1 kita telah mendapatkan visualisasi graf dari masalah yang ingin kita selesaikan. Untuk menyelesaikan masalah pengurutan ini kita akan menggunakan langkah-langkah seperti pada bagian II pada makalah ini.

Dari gambar kita bisa melihat, bahwa simpul/pekerjaan yang tidak membutuhkan prerequisite atau ditunjuk atau pekerjaan lain adalah pekerjaan A dan pekerjaan I. Untuk salah satu alternatif pengurutan, misal kita mengambil pekerjaan A, sebagai pekerjaan yang akan pertama dilakukan. Pekerjaan A ini kita masukkan kedalam list L kita yang masih kosong, sehingga anggota L sekarang adalah {A}. Selanjutnya kita perlu menghapus simpul A dari graf pada gambar 3.1 sehingga didapat hasil sebagai berikut.



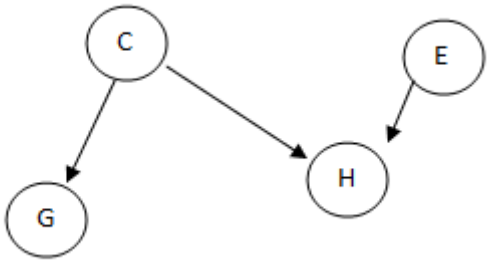
Gambar 3.2 Graf setelah simpul A dihapus

Sekarang kita memiliki graf baru tanpa simpul A didalamnya. Pada gambar kita bisa melihat bahwa simpul D tidak lagi ditunjuk oleh simpul lain manapun. Sehingga sekarang kita memiliki dua buah simpul yang bisa dihapus, yaitu simpul I dan simpul D. Kali ini kita akan memilih untuk menghapus simpul I terlebih dahulu, dan memasukkannya sebagai elemen terakhir dalam list L. Sehingga anggota list L sekarang adalah {A,I}. Setelah menghapus simpul A dari graf, maka graf kita sekarang akan menjadi sebagai berikut :



Gambar 3.3 Graf setelah simpul A dan I dihapus

Sekarang kita simpul B dan simpul E juga menjadi simpul tanpa prerequisite (tidak ditunjuk oleh simpul lainnya). Jadi, simpul yang bisa dihapus adalah simpul B,E dan D. Jika kita menghapus simpul D dan B secara berurutan, maka list kita sekarang akan beranggotakan {A,I,D,B}. Kemudian graf kita tinggal menyisakan 4 simpul sebagai berikut :



Gambar 3.3 Graf setelah simpul A,I,D dan B dihapus

Selanjutnya kita dapat menghapus simpul C dan simpul E dari dalam graf. Jika kita menghapus C dan E secara berurutan maka graf kita akan menjadi sebagai berikut :

Gambar 3.4 Graf setelah simpul A,I,D,B,C dan E dihapus

Anggota list L kita sekarang adalah {A,I,D,B,C,E}. Dari gambar diatas kita bisa melihat bahwa graf kita tinggal menyisakan dua buah simpul yaitu simpul G dan H yang tidak terhubung satu sama lain. Oleh karena itu kita bisa menghapus kedua simpul tersebut. Jika kita menghapus simpul H kemudian menghapus simpul G, maka anggota graf kita sekarang akan menjadi kosong dan kita akan beranggotakan {A,I,D,B,C,E,H,G}.

Anggota-anggota list yang kita dapatkan merupakan salah satu alternatif urutan-urutan pekerjaan yang bisa kita lakukan. Jadi, dari hasil ini, urutan pekerjaan yang bisa kita lakukan berturut-turut yaitu pekerjaan A, I, D, B,

C, E, H, dan G.

Alternatif jawaban yang lain bisa didapat jika kita mengambil keputusan berbeda saat ada dua atau lebih simpul yang bisa dihapus. Sebagai simpul yang pertama bisa dihapus adalah A dan I. Jika kita menghapus simpul I terlebih dahulu, tentu urutan pekerjaan yang akan kita dapat akan berbeda dimana pekerjaan I akan dikerjakan lebih dulu daripada pekerjaan-pekerjaan lainnya.

IV. KESIMPULAN

Teorema graf memiliki banyak sekali aplikasi terapannya. Salah satu contohnya adalah dalam menyelesaikan topological sorting. Menggunakan sebuah Directed Acyclic Graph, persoalan topological sort ini bisa divisualisasikan dengan sangat baik. Algoritma penyelesaiannya pun tidak terlalu rumit dan cukup mudah untuk dipahami. Solusi dari sebuah kasus topological sorting tidaklah unik, terdapat beberapa alternatif jawaban karena perbedaan pengambilan keputusan saat akan menghapus dua atau lebih simpul yang bisa dihapus.

REFERENSI.

[1] R. Munir, *Diktat Kuliah IF2091 Struktur Diskrit.* Bandung: Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung, 2008.

[2] Liem Inggriani, *Diktat Struktur Data*. Bandung : Penerbit ITB.

[3] http://en.wikipedia.org/wiki/Topological_sorting, diakses pada tanggal 10 desember pukul 13.10 WIB

[4] http://id.wikipedia.org/wiki/Teori_graf, diakses pada tanggal 10 desember pukul 14.05 WIB

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

Muhammad Afif Al-hawari (13510020)