Aplikasi Graf dalam Perancangan Program

Nursyahrina - 13513060
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
nursyahrina17@gmail.com

***Abstract*— Graf adalah gabungan dari himpunan tak-kosong dari simpul dengan himpunan sisi. Simpul merepresentasikan sebuah objek dan sisi merepresentasikan hubungan dari dua objek. Teori graf dapat digunakan dalam kehidupan sehari-hari, diantaranya dalam topologi jaringan, penjadwalan, pewarnaan peta, pencarian rute terpendek untuk mengatasi permasalahan tukang pos dan pedagang keliling dan lain-lain. Pada makalah ini akan dibahas salah satu bentuk lain dari penggunaan graf dalam kehidupan sehari-hari yaitu pada proses perancangan program dalam konteks prosedural.**

***Keywords*—algoritma, graf, program.**

I. PENDAHULUAN

Ditengah pesatnya perkembangan zaman, manusia sudah terbiasa berinterkasi dengan kecanggihan teknologi yang ada. perkembangan teknologi yang paling terasa dan sangat berpengaruh pada saat ini adalah perkembangan teknologi gadget. engineer semakin berpacu membuat inovasi-inovasi baru yang hampir sulit dipahami oleh kebanyakan orang awam. Bermunculan bermacam software dan aplikasi yang unik dengan fungsionalitas yang tinggi. Karena itu banyak orang dari berbagai kalangan yang tertarik untuk mempelajari hal-hal yang ber'bau' teknologi, dimulai dari ilmu tentang pemrograman.

Ilmu dan teknik pemrograman telah tersebar luas dan diminati oleh berbagai kalangan. Baik belajar langsung diperguruan tinggi, maupun belajar melalui media lain seperti kursus atau belajar secara otodidak. Membahas tentang pemrograman, tidak hanya sebatas membuat kode program (*coding*), namun juga bagaimana membuat algoritma yang tepat dan efisien dari segi kompleksitas (pemakaian waktu) dan dari segi memori. Program seharusnya dirancang dengan teratur dan sesuai tahapan. Perancangan awal ini juga dimaksudkan untuk memperjelas konsep awal dalam pembuatan program. Oleh karena itu, untuk menghasilkan program dengan algoritma yang tepat dan efisien, dibutuhkan media yang baik dalam perancangannya. Dalam perancangan program dapat digunakan diagram-diagram implementasi dari graf sebagai representasi dari rancangan algoritma yang sesungguhnya.

II. DASAR TEORI

A. Definisi Graf

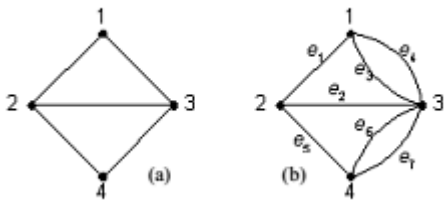
Graf adalah gabungan dari himpunan tak-kosong dari simpul dengan himpunan sisi. Simpul merepresentasikan sebuah objek dan sisi merepresentasikan hubungan dari dua objek. Secara matematis, graf G didefinisikan dengan gabungan himpunan. Graf $G = (V, E)$. V adalah himpunan tidak-kosong dari simpul-simpul = {v1, v2, ..., vn } dan E adalah himpunan sisi (*edges*) yang menghubungkan dua simpul = {e1, e2, ..., en}. e1 adalah sisi yang menghubungkan simpul v1 dan v2, maka e1 dinotasikan dengan e1=(v1,v2).
Definisi diatas menyebutkan bahwa graf bisa tidak memiliki sisi dan hanya memiliki simpul. Graf yang hanya punya simpul dan tidak mempunyai sisi, maka graf tersebut disebut graf kosong.

B. Jenis-Jenis Graf

- Berdasarkan ada/tidaknya gelang atau sisi ganda dalam suatu graf, graf dibedakan menjadi dua jenis.

1. Graf sederhana (*simple graph*) . Graf sederhana adalah graf yang tidak memilki gelang maupun sisi ganda.

2. Graf tak-sederhana (*unsimple graph*)
Graf tak-sederhana adalah graf yang mengandung gelang atau sisi ganda.

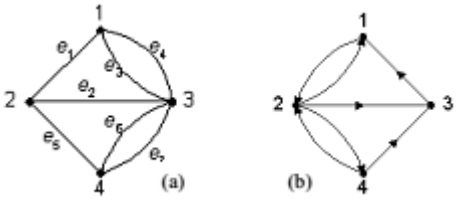


Gambar 2.1 Graf sederhana (a), dan Graf tak-sederhana (b)
[1] source: Munir, Rinaldi. 2008. "Diktat Kuliah IF 2091 Struktur Diskrit". Bandung:Program Studi Teknik Informatika STEI ITB.

- Berdasarkan orientasi arah pada sisi-sisinya, graf dibedakan menjadi dua jenis:

1. Graf tak-berarah (*undirected graph*)
Graf tak berarah adalah graf yang sisi-sisinya tidak memiliki orientasi arah.

2. Graf berarah (*directed graph*/*digraph*)
Graf berarah adalah graf yang sisinya memiliki orientasi arah.



Gambar 2.2 Graf tak-berarah (a), dan Graf berarah(b)
[1] source: Munir, Rinaldi. 2008. "Diktat Kuliah IF 2091 Struktur Diskrit". Bandung:Program Studi Teknik Informatika STEI ITB.

C. Terminologi Graf

1. Ketetanggaan(*Adjacent*)
Dua buah simpul dikatakan bertetangga jika keduanya terhubung oleh satu atau lebih sisi.

2. Bersisian (*Incidency*)
Sisi e dan simpul v dikatakan bersisian jika simpul v terhubung dengan simpul lain melalui sisi e.

3. Simpul Terpencil (*Isolated Vertex*)
Simpul terpencil adalah simpul yang tidak bersisian dengan sisi manapun di dalam graf.

4. Graf Kosong (*Null Graph* atau *Empty Graph*)
Graf kosong adalah graf yang himpunan sisinya adalah himpunan kosong.

5. Derajat (*Degree*)
Derajat suatu simpul adalah sama dengan jumlah sisi yang bersisian dengan simpul tersebut.

6. Lintasan (Path)
Lintasan adalah barisan berselang-seling antara simpul dan sisi yang berbentuk v0, e1, v1, e2, v2, …, en, vn sedemikian sehingga e1 = (v0,v1), e2 = (v1,v2) dan seterusnya adalah sisi-sisi dari graf G.

7. Terhubung (*Connected*)
Dua buah simpul v1 dan v2 dikatakan terhubung jika ada lintasan dari v1 menuju ke v2 ataupun sebaliknya.

D. Graf Khusus

1. Graf Lengkap (*Complete Graph*)
Graf lengkap adalah graf sederhana yang setiap simpulnya memiliki sisi menuju semua simpul yang lain. Jika G adalah graf dengan n buah simpul, maka setiap simpulnya bersisian dengan n-1 buah sisi.

2. Graf Lingkaran (*Cyrcle Graph*)
Graf lingkaran adalah graf sederhana yang setiap simpulnya memiliki derajat dua (setiap simpulnya hanya bersisian dengan dua buah sisi).

3. Graf Teratur (*Regular Graph*)
Graf teratur adalah graf yang setiap simpulnya memiliki derajat yang sama.

III. ALGORITMA DAN PROGRAM

A. Definisi Algoritma

Algoritma adalah urutan langkah-langkah logis penyeselaian masalah yang disusun secara sistematis dan logis. Dalam merancang algoritma, ada tiga hal yang harus diperhatikan. Algoritma yang dihasilkan haruslah benar (logis). Selain itu, algoritma tersebut juga harus memberikan hasil yang benar atau paling mendekati hasil yang seharusnya. Hal terakhir yang harus diperhatikan adalah efisiensi algoritma tersebut, baik dari segi kompleksitas maupun dari segi memori yang digunakan.

B. Sejarah Penemuan Algoritma

Para ahli sejarah matematika menemukan asal kata algoritma berasal dari nama penulis buku arab yang terkenal yaitu Abu Ja'far Muhammad Ibnu Musa Al-Khuwarizmi. Al-Khuwarizmi dibaca orang barat menjadi Algorism. Al-Khuwarizmi menulis buku yang berjudul Kitab Al Jabar Wal-Muqabala yang artinya "Buku pemugaran dan pengurangan" (The book of restoration and reduction). Dari judul buku itu kita juga memperoleh akar kata "Aljabar" (Algebra). Perubahan kata dari algorism menjadi algorithm muncul karena kata algorism sering dikelirukan dengan arithmetic, sehingga akhiran –sm berubah menjadi –thm. Karena perhitungan dengan angka Arab sudah menjadi hal yang biasa, maka lambat laun kataalgorithm berangsur-angsur dipakai sebagai metode perhitungan (komputasi) secara umum, sehingga kehilangan makna kata aslinya. Dalam bahasa Indonesia, kata algorithm diserap menjadi algoritma.[1]

C. Algoritma dan Program

Algoritma dan program adalah dua hal yang berbeda. Program adalah kumpulan instruksi komputer yang dibuat dengan bahasa yang dikenali komputer, sementara tahapan atau langkah-langkah sistematis penyusun program adalah algoritma. Program dibuat dengan bahasa yang dikenali oleh komputer, sedangkan dalam perancangan algoritma, kita bebas menggunakan bahasa apa pun, namun diutamakan menggunakan notasi algoritmik karena dalam

tahapan pembuatan program selanjutnya, notasi algoritmik dapat ditranslasikan ke berbagai bentuk bahasa pemrograman.

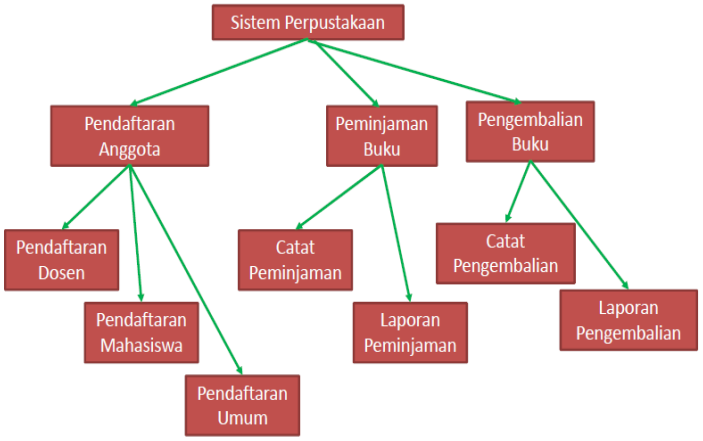
IV. PERANCANGAN PROGRAM

A. Perancangan Algoritma

Dalam perancangan algoritma maupun program, tahapan awal yang harus dilakukan adalah dekomposisi persoalan. Dalam tahapan ini, permasalahan yang besar dipecah-pecah ke bentuk persoalan-persoalan yang lebih kecil dan rinci. Dekomposisi dilakukan untuk membagi pekerjaan agar lebih jelas, teratur dan hemat waktu karena masing-masing permasalahan dapat dikerjakan secara paralel.

Setelah melalui tahapan dekomposisi, algoritma yang dibentuk akan lebih sederhana karena telah dipecah menjadi beberapa subprogram dalam bentuk fungsi dan prosedur. Setiap prosedur juga memanggil prosedur yang lain. Algoritma yang dihasilkan lebih efisien karena menghindari penulisan algoritma yang sama berulang-ulang.

Contoh dekomposisi masalah adalah sebagai berikut:



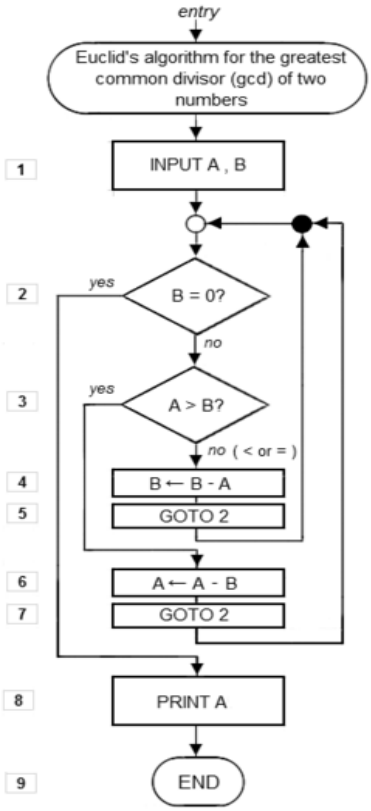
Gambar 4.1 Dekomposisi untuk Sistem Perpustakaan
Sumber : Slide Kuliah IF1208_08_DekomposisiMasalahdanNotasiAlgoritmik.pdf

---

Dekomposisi sederhananya dapat direpresentasikan dengan graf, seperti gambar diatas. Rancangan Algoritma untuk sistem perpustakaan dipecah menjadi tiga bagian , yaitu pendaftaran anggota, peminjaman buku dan pengembalian buku. Pendaftaran dapat dilakukan oleh dosen, mahasiswa dan pengunjung umum. Pada masing-masing proses peminjaman dan pengembalian buku dilakukan pencatatan dan pelaporan. Perancangan algoritma biasanya dilakukan dengan menggunakan diagram alir *(flow chart)*. Flow chart ini merupakan salah satu implementasi dari graf berarah.

Contohnya, algoritma *Euclid.* Algoritma ini bertujuan untuk menentukan PBB (Pembagi Bersama Terbesar) dari dua buah bilangan bulat. Jika diberikan dua bilangan bulat tak-negatif A dan B, berikut adalah algoritma *Euclid* untuk menentukan PBB(A,B).

Gambar disamping adalah flow chart dari algoritma *Euclide*.

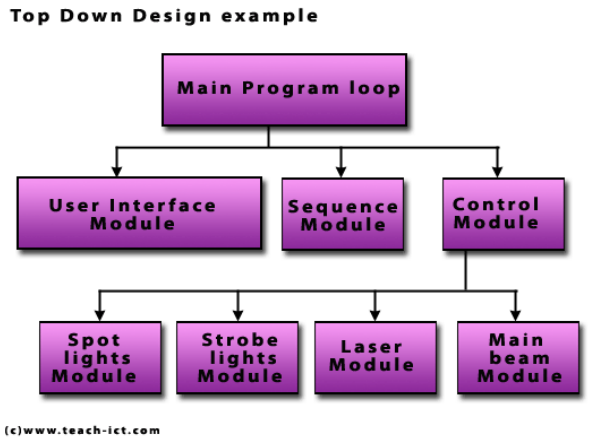


Gambar 4.2 *Flow Chart* Algoritma Euclid
Sumber : https://mhs.blog.ui.ac.id/ghazian.zhafiri/2014/02/26/algorithm-and-programming-language/ (diakses pada 11 Desember 2014)

B. Modularisasi Program

Dalam pengerjaan program skala menengah sampai skala besar, diperlukan modularisasi program. Prinsipnya hampir sama seperti dekomposisi. Program akan petakan ke dalam modul-modul dan dikerjakan secara terpisah.

Pembagian program menjadi beberapa modul mempermudah pengerjaan secara team work. Modul-modul juga dapat dites secara terpisah. Modifikasi program juga akan lebih mudah dilakukan karena hanya melibatkan modul tertentu saja. Implementasi modul dapat berupa fungsi, prosedur, ADT(*Abstract Data Type*) dan kelas.



Gambar 3.1 Pohon untuk modular program
Sumber : http://www.teach-ict.com/as_as_computing/ocr/H447/F453/3_3_7/programming/miniweb/pg2.htm (diakses pada 11 Desember 2014)

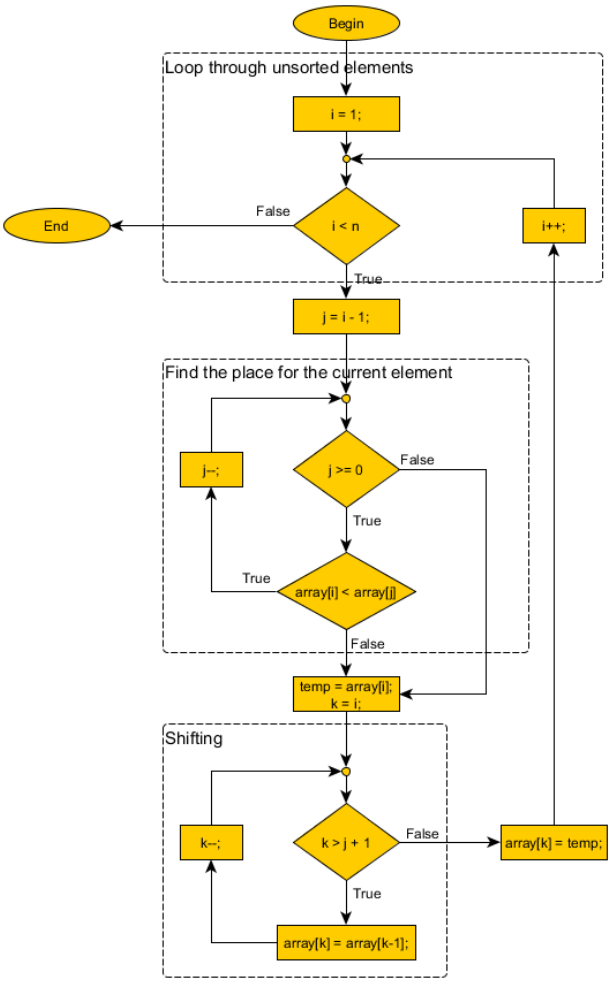
Seperti pada contoh diatas, pemodulan program dapat dilakukan dengan graf berarah tanpa sirkuit atau bisa disebut juga dengan Pohon. Simpul-simpul pada graf diatas, V = *{Main Program Loop, User Interface Module, Sequence Module, Control Module, Spot Lights Module, Strobe Lights Module, Laser Module, Main Beam Module}*. Modul –modul tersebut dibuat terpisah namun dapat memanggil modul yang lainnya.

Proses looping pada main program bercabang 3 (*case*) memanggil *User Interface Module, Sequence Module* dan *Control Module*. Jika *main program* kemudian memanggil *Control Module*, maka proses selanjutnya akan beralih ke simpul-simpul selanjutnya(*Spot Lights Module,Strobe Lights Modules, Laser Modul*, dll

V. MERANCANG ALGORITMA

Berikut adalah contoh cara merancang algortima dengan representasi graf. Algoritma yang akan dibuat graf adalah algoritma *insertion sorting* pada array integer. Idenya adalah melakukan pengurutan sambil menyisipkan elemen yang sudah terurut. Mencari tempat yang tepat untuk menyisipkan elemen agar keseluruhan elemen array terurut. Prosesnya akan dilakukan sebanyak n-1 kali dengan n adalah jumlah elemen pada array. Penyisipan dilakukan dengan menggeser elemen-elemen yang masih belum terurut.

Algoritma ini jika dibuat dengan representasi graf makan akan menghasilkan graf sebagai berikut :



Gambar 3.1 Graf (*Flow Chart) Insertion Sorting*
Sumber : http://www.c-programming-simple-steps.com/insertion-sort.html (diakses pada 11 Desember 2014)

Gambar diatas menunjukkan representasi graf dari algoritma *insertion sorting*. Array pada kondisi pensortiran terbagi dua (tidak dibagi secara langsung namun tatp dalam array yang sama). Bagian yang sudah diurut dan yang belum diurutkan. Pada bagian yang belum

terurut dilakukan pengecekan nilai, mencari nilai elemen yang menyalahi pngurutan kemudan mencari tempat yang teapt untuk menyisipkan elemen tersebut.

Bagian *Shifting* pada graf diatas adalah cara untuk menyisipkan elemen dengan terlebih dahulu menggeser elemen lainnya. Kemudian proses yang sama dilakukan terus menerus (*looping*) sampai pengurutan selesai.

Sorting juga dapat dilakukan dengan berbagai algoritma sorting yang lain seperti *counting sort*, *selection sort*, *bubble sort*, dll. Algoritma ynag berbeda akan menghasilkan graf yang berbeda pula.

VI. SIMPULAN

Graf diaplikasikan dalam banyak bidang ilmu pengetahuan termasuk salah satunya dalam bidang informatika. Perancangan program dapat diawali dengan graf sebagai representasi dari algoritma sesungguhnya. Perancangan awal ini juga dimaksudkan untuk memperjelas konsep dan gambaran program secara keseluruhan. Perancangan direpresentasikan dengan graf berarah. Tahapan perancangan ini salah satu cara untuk meningkatkan efisiensi waktu dan efektifitas kerja program. selain untuk perancangan awal, graf berarah juga dapat digunakan sebagai representasi dari modularisasi program dalam skala menengah dan besar. Penggunaan graf dalam hal ini memudahkan dalam pembagian modul saat kerja tim.

REFERENSI

[2] https://andikafisma.wordpress.com/algoritma-dan pemrograman/ (diakses pada 11 Desember 2014)

[3] Munir, Rinaldi. 2008. "Diktat Kuliah IF 2091 Struktur Diskrit". Bandung:Program Studi Teknik Informatika STEI ITB.

[4] Tim Pengajar IF1210 Dasar Pemberograman.2014. Slide Kuliah Modular Programming

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, 10 Desember 2014

NURSYAHRINA (13513060)