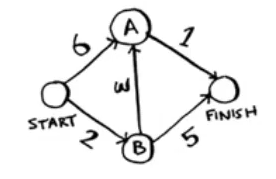
Konsep Algortima Djikstra

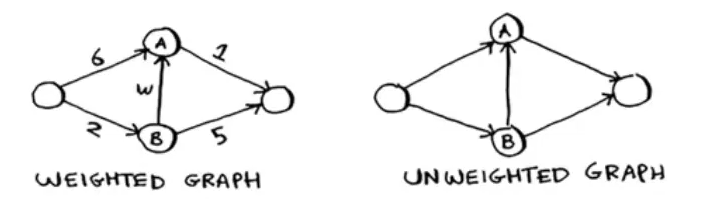
* Kumpulan node (simpul) yang membentuk graf berbobot dan berarah yang dimulai dari simpul awal sampai simpul tujuan dalam mencari rute terpendek
* Djikstra Algoritma adalah algortima greedy yang dipakai dalam memecahkan permasalahan jarak terpendek untuk sebuah graf berarah dengan bobot sisi(edge) yang bernilai tak negative
* Djikstra Algoritma hanya berkerja/digunakan pada Directed Acyclic Graph atau biasa disingkat DAG

Cara Kerja Djikstra Algoritma

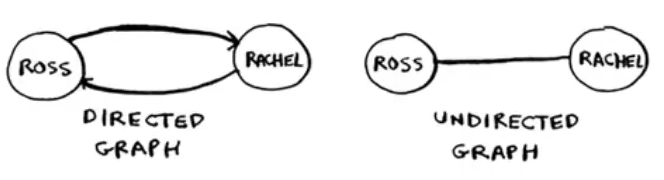
* Disni kita mempunyai 4 node : star node,A node,B node,dan finish node
* Langkah pertama pilih antara node A dan B,kita compare mana yang lebih kecil
  + Karena kita belum tau berapa yang dibutuhkan untuk ke node finish maka menuju node finish kita beri infinity
  + Disini kita tau bahwa node B lebih dekat dengan finish setelah kita compare dengan node A
* Langkah kedua seperti langkah diatas kita telah memilih node B sebagai node terkecil dibandingkan denga A
  + Di node B kita melakukan update terhadap semua neighboors dari node B,neighboors dari node B adalah node A dengan jumlah 3,sedangkan dari star menuju B adalah 2 berarti dari node star sampai node A melalui node B memiliki jumlah 5, dari data sebelumnya bahwa dari node star sampai A adalah 6 berarti bahwa data yang kita dapatkan barusan lebih baik dari data sebelunya
  + Ada satau neighboors dari node B yaitu node finish,pada data sebelumnya kita belum tau dari node star menuju node finish perlu jumlah berapa atau tadi kita kasih nila infinity,tetapi disini kita sudah tau bahwa nilainya 7 dari jalur star,B,finish
* Langka ke tiga adalah kita mengulangi lagi langkah pertama
  + Disini kita mengcompre lagi dari node A yang melalui node B dengan node finish,didapatkan bahwa node A lebih kecil dari node finish maka kita pilih yang node A
  + Setelah kiita memilih nnode A neigboors dari node A adalah node finish dengan biaya 1
  + Dari data sebelumnya kita tau bahwa node A adalah 5 dan dari node A sampai finish adalah 1,jadi total nya adalah 6,sedangkan dari data yang tadi kita dapatkan bahwa untuk sampai ke node finish membutuhkan 7 artinya data yang baru saja kita dapat lebih baik dari data sebelumnya

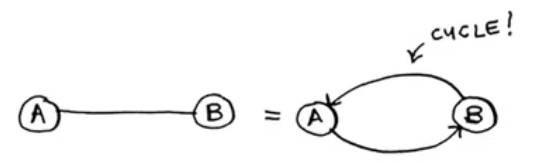
Directed Acyclic Graph

* DAG adalah graph berarah yang tidak memiiki cycle/loop didalamnya
* Kompenen DAG adalah root dan leaf
* Root adalah simpul tanpa sisi masukan
* Leaf adalah simpul tanpa sisi keluar



* Weighted Graph adalah graph yang memiliki bobot
* Unweighted Graph adalah graph yang tidak memiliki bobot
* Untuk mencari jalur terpendek pada weighted graph kita menggunakan Djikstra Algoritma
* Sedankan pada unweighted graph kita menggunakan BFS



* Directed Graph adalah graph yang memiliki anak panah
* Sedangan undirected graph adalah graph yang tidak memiliki anak panah
* Undirected grah sama dengan cycle graph yang artinya a bisa ke b dan b bisa ke a
* Seperti gambar dibawah

Recap

* Djikstra algoritma hanya berkerja pada positive weights
* Jika kita menemui negative weights kitab isa menggunakan Bellman-Ford algortima
* BFS digunaan untuk mencari shortest path pada unweighted graph
* Djikstra algoritma digunakan untuk mencari shortest path pada weighted graph

Konsep Algoritma Greddy

* Algoritma greedy adalah pendekatan dalam pemrograma yang memilih langkah terbaik pada setiap langkah dalam harapan bahwa itu akn mengarah pada solusi global terbaik.
* Pada setiap langkah,algoritma greedy memilih opsi yang tampaknya paling menguntungkan pada saat itu,tanpamempertimbangkan implikasi jangka Panjang atau memeriksa solusi alternatif

Masalah yang dapat diselesaikan dengan Algoritma Greedy

* Masalah pemilihan Aktivitas (Activity Selction Problem)
* Masalah Pemotongan Batang (Cutting Stock Problem)
* Masalah Koin Minimum (Minimum Coin Change Problem)
* Masalah Pencocokan Interval (interval Scheduling Problem)
* Masalah Pohon Merentang Minimum (Minimum Spaning Tree Problem)
* Masalah Ransel (Knapsack Problem)
* Masalah Pemilihan Jalur Terpendek (Shortest Path Problem)
* Masalah Penjadwalan Tugas (Task Scheduling Problem)

Klasifikasi Algoritma Greedy

* Greedy Terikat (Bound Greedy)
  + Mempertimbangkan Batasan atau kriteria tertentu dalam pengambilan keputusan greedy
  + Memilih langkah terbaik yang memenuhi Batasan yang ditetapkan
  + Pilihan yang dibuat tidak melampaui batasan atau kriteria yang telah ditentukan sebelumnya
* Greedy Tidak Terikat (Unbound Greedy)
  + Tidak mempertimbangkan Batasan atau kriteria tertentu dalam pengambilan keputusan serakah
  + Memilih langkah terbaik berdasarkan factor factor local tanpa mempertimbangkan Batasan global atau kriteria yang ditetapkan sebelumnya
* Jenis Algoritma Greedy
  + Greedy Terikat (Bound Greedy)
    - Algoritma Prim
    - Algoritma Kruskal
    - Algoritma Set Cover
    - Algoritma Job Scheduling
    - Algoritma Huffman Coding
  + Greedy Tidak Terikat (Unbound Greedy)
    - Algoritma Djikstra
    - Algoritma Knapsack Fractional
* Kesimpulan
  + Algoritma Greedy merupakan algoritma yang sangat berguna dalam menyelesaikan masalah optimasi
  + Algoritma ini berkerja dengan memilih opsi terbaik pada setiap tahap,sehingga mencapai solusi terbaik pada akhirnya
* **Knapsack Problem**
  + Diterjemahkan menjadi ransel/karung
  + Permasalahan Pengelola Ruang
  + Ransel sebagai ruang penyimpanan barang
  + Spesifikasi Barang : Berat,Kegunaan,Ukuran
  + Spesifikasi Ransel : Kapasitas/Daya Tampung
* **Penyelesian Knapsack dengan Greedy**
  + Fokus pada 3 kondisi Prioritas
    - Profit lebih besar
    - Berat lebih kecil
    - Perbandingan Profit/Berat lebih besar
  + Ketiga di atas disebut Optimal Solution
* **Case Scheduling**
  + Misalkan kita mejadwalkan suatu kelas maka langkah yang harus dilakukan adalah
    - Cari kelas berakhiran paling awal
    - Cari kelas setelah kelas pertama berakhir lalu pilih lagi kelas yang paling luar
* **Set Covering Problem**
  + Misalkan kita mempunyai program radio langkah yang harus kita lakukan adalah
    - Memilih stasiun radio dengan jangkuan paling luas tidak masalah jika stasion ini telah di jangkau sebelumnya
    - Ulangi langka ini sampai stasion tercover keseluruhan
    - Ini biasa disebut dengan Algoritma Approximmation
      * Tidak mencari solusi paling optimal tapi mendekati optimal
      * Approximmation banyak digunakan ketika menyelesaikan problem yang membutuhkan waktu yang sangat lama