Fakultas Informatika
School of Computing
Telkom University
Think Big! Go Forward!

Telkom University

ANALISIS ALGORITMA

Week 07: Problem Komputasi Graf

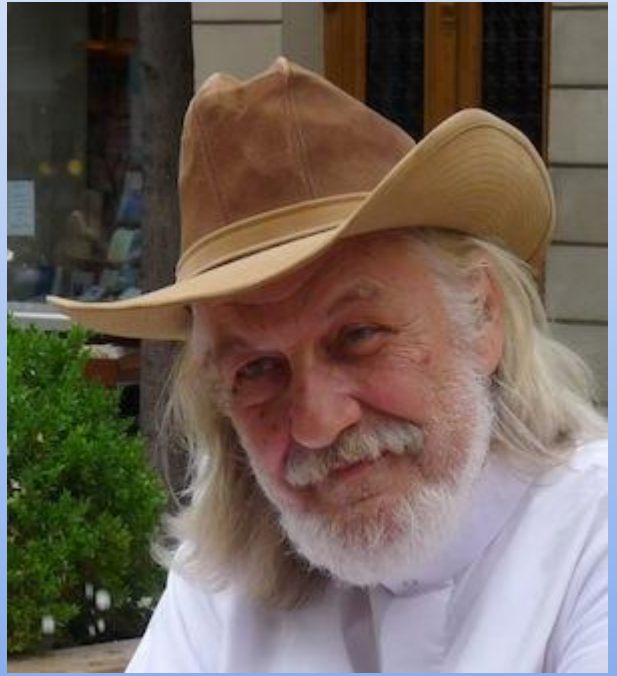
PROGRAM PASCA SARJANA INFORMATIKA
FAKULTAS TEKNIK INFORMATIKA
UNIVERSITAS TELKOM

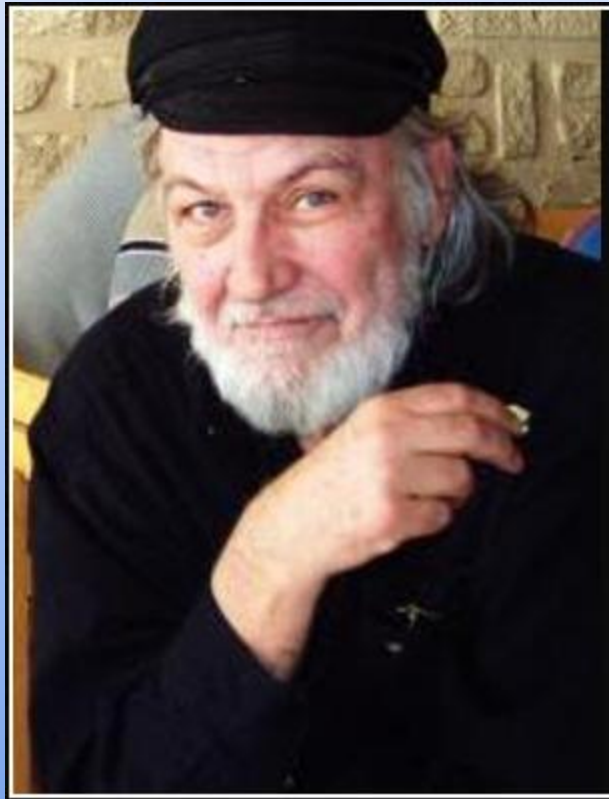
2022/2023

Problem Menarik Lainnya

Matching / Pasangan

- Diberikan graf (tidak berarah) G=(V, E)
- Cari himp. terbesar edges M subset E,
  - (u,v) dan (x,y) di M: u == x iff v == y
- Mudah dicari untuk graf BipartIt
- Mungkin untuk graf lain
  - Jack Edmonds (April 5, 1934) in 1965
  - Algoritma Blossom
- Matching Maximum
- Matching Sempurna (Perfect)

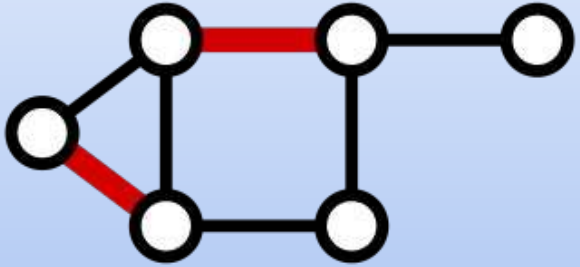


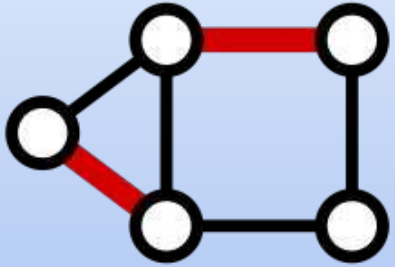


The classes of problems which are respectively known and not known to have good algorithms are of great theoretical interest. [...] I conjecture that there is no good algorithm for the traveling salesman problem. My reasons are the same as for any mathematical conjecture: (1) It is a legitimate mathematical possibility, and (2) I do not know.

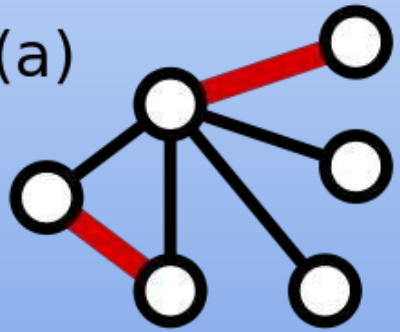
— Jack Edmonds —

AZ QUOTES

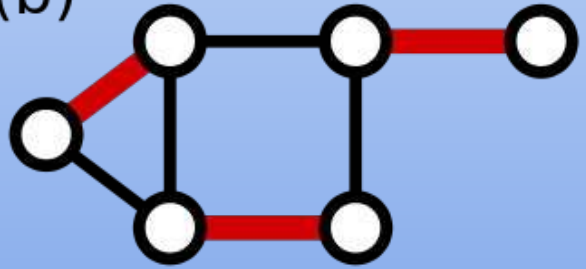




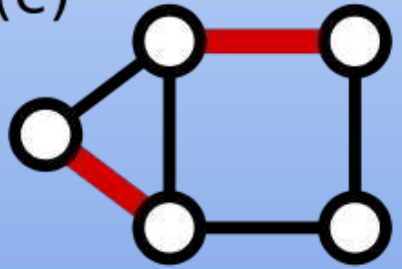
(a)



(b)



(c)



Alternating Path dan Augmenting Path

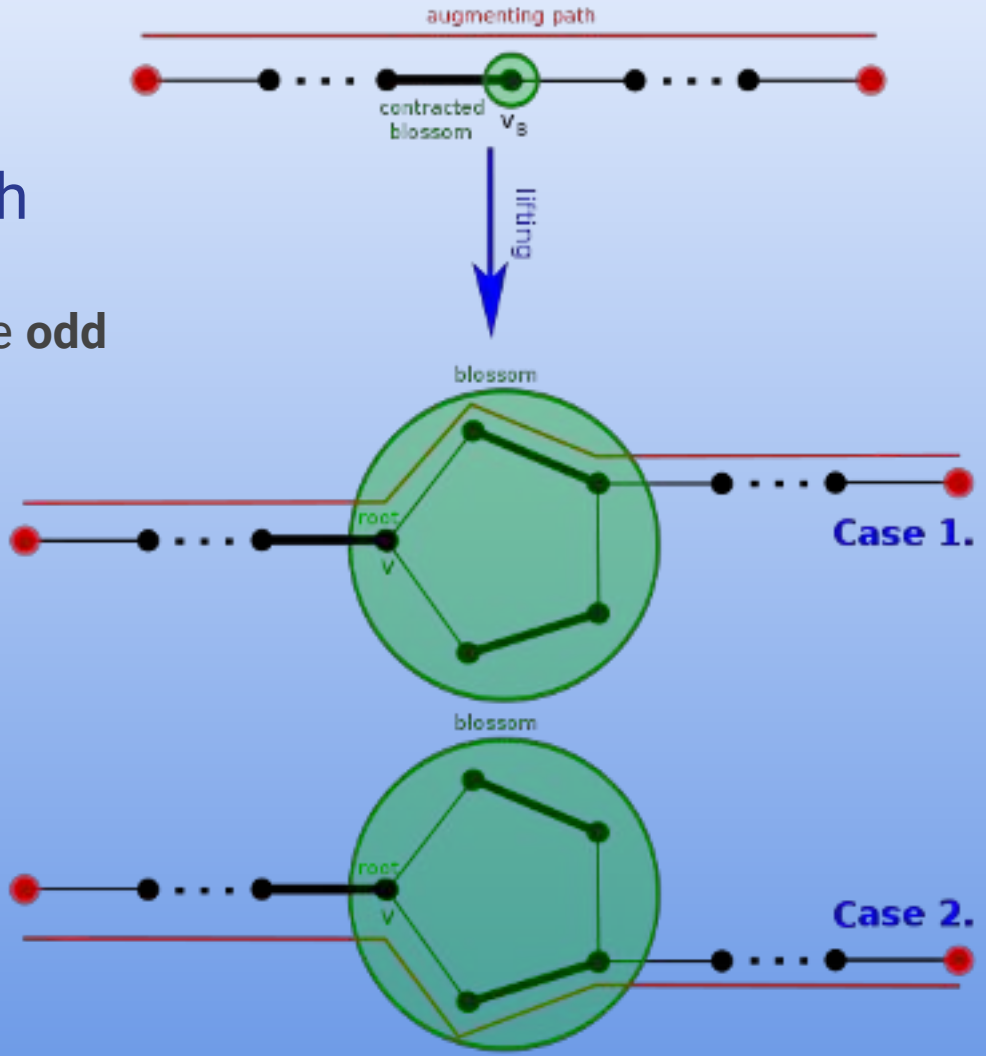
- Satu pasang (match) adalah edge dalam graf
  a. Tidak ada pasangan yang berbagi verteks
- Maximal matching tidak mempunyai calon pasangan lagi
- Maximum matching mempunyai pasangan terbanyak dari semua kemungkinan maximal matchings
- Perfect matching tidak menyisakan verteks tidak berpasangan
- Alternating path: jalur selang-seling antara edges berpasangan dan tidak
  a. Boleh mulai dari dan berakhir pada verteks belum berpasangan
  b. Boleh mulai dari dan berakhir pada verteks yang sudah dipasangkan
  c. Boleh mulai dari verteks yang belum dan berakhir di veteks yang sudah berpasangan, vv
  d. Alternating loop, dimana verteks awal dan akhir sama
- Augmenting path adalah alternating path yang (a)
- Alternating loop in bipartite graph is always even

Matching on Bipartite Graph

- Ganti edge pasangan dengan alternatifnya pada augmenting path akan menghasilkan pasangan yang lebih banyak
- M dan P merupakan himpunan matching (edges) dan augmenting path
- Dan M⊕P himpunan edges yang tidak berbagi antara M dan P
  - i.e. xor operation; $M \oplus P = (M \cup P) - (M \cap P)$
- M maximum matching **iff** tidak ada lagi augmenting path

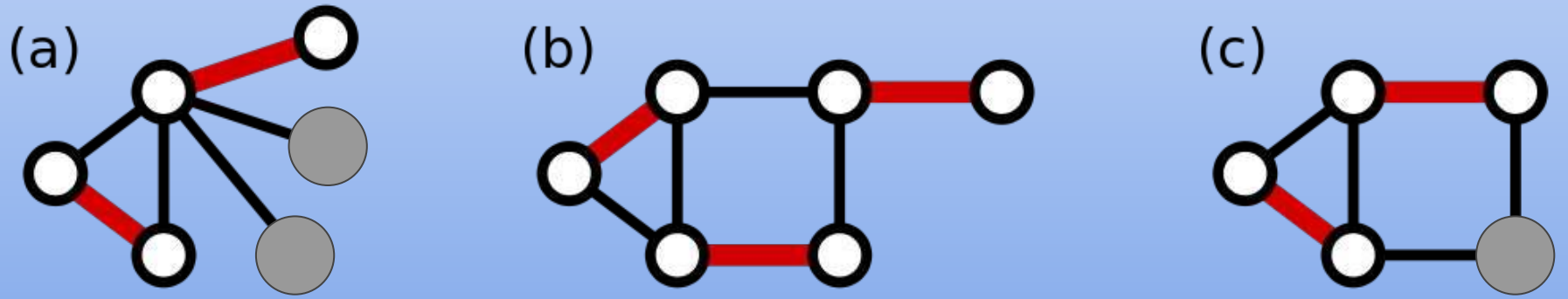
Matching untuk General Graph

Alternating loop in general graph can be **odd**



Edge Cover

- Diberikan graf G=(V, E)
- Cari himpunan terkecil C $\subseteq$ E, $\forall v \in$ V, maka (v,x) atau (x,v) termasuk C



Edge Cover

- Preprocessing dengan menggunakan algoritma Matching graf umum
- Tambahkan edges menghubungkan verteks yang belum ter-cover
- Buktikan bahwa algoritma diatas akan menghasilkan edge cover minimum
  - Dengan kontradiksi

Vertex Cover dan Problem Terkait

Vertex Cover

- Diberikan graf G=(V, E)
- Cari himpunan terkecil C $\subseteq$ V, dengan $\forall (u,v) \in$ E, maka u atau v atau keduanya ada di C
- Minimal VC (⇨)
- Minimum VC (⇓)

Vertex Cover untuk Bipartite

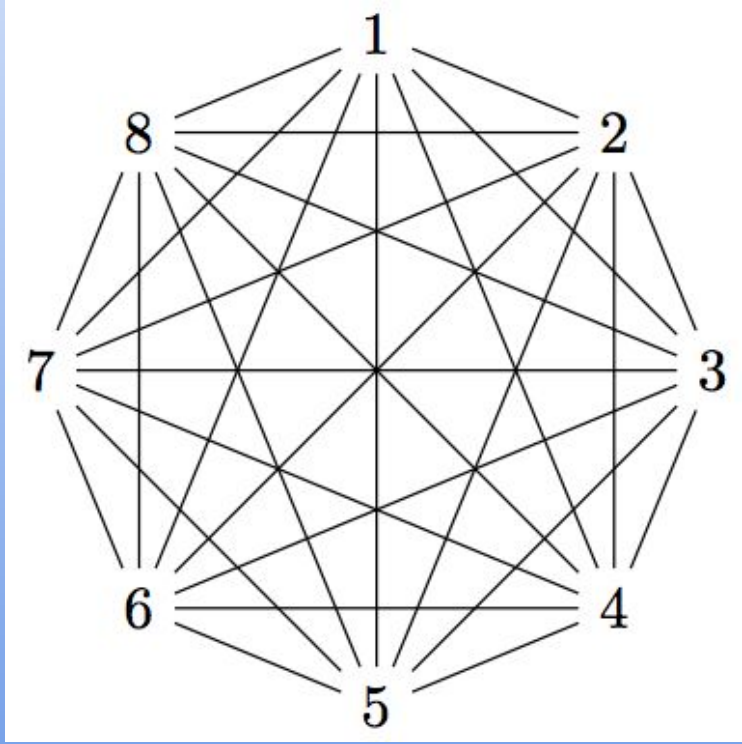
- Gunakan solusi Matching untuk graf Bipartite
- Buat alternative paths dari semua verteks K yg belum berpasangan
  - Masukan semua pasangan verteksnya di L kedalam himpunan Vertex Cover C
- Buat alternative paths dari semua verteks L yg belum berpasangan
  - Masukan semua pasangan verteksnya di K kedalam himpunan Vertex Cover C
- Untuk setiap edges berpasangan yang belum digunakan diatas
  - Ambil semua verteks di K (atau boleh juga yg di L) untuk masuk ke Vertex Cover C
- Buktikan (dengan Kontradiksi)
  - Misalkan vertex cover hasil C tidak minimum (yaitu C hanya minimal)
  - Maka, ada vertex cover lain C' (dengan algoritma lain), dimana |C'| < |C|
  - Ambil c' subset dari C' yang tidak berinterseksi dengan C, begitu juga untuk c
  - Sebuah edge (u,v) harus ter-cover oleh u, v, atau keduanya.
  - dst ...

Vertex Cover u/ Stars, Trees, dan Forest

- Mulai dari leaves (dengan degree 1)
- Setiap bapak dari leaves parents dimasukkan ke Vertex Cover VC_T
  - Hapus semua leaves dan bapaknya dari graf
- Ulangi proses tersebut sampai semua edges habis diproses
- Juga buktikan hasil minimum vertex cover (dengan kontradiksi)
  - Misalkan vertex cover VC^* dengan cara lain menghasilkan himpunan yang lebih kecil
  - Artinya ada beberapa verteks di VC_T yang dapat digantikan oleh lebih sedikit verteks di VC^*
  - Verteks pengganti dapat berupa leaves atau non-leaves
  - … akibatnya… dst…
- Bagaimana jika strategi adalah justru dimulai dari verteks dengan derajat terbesar lebih dulu?

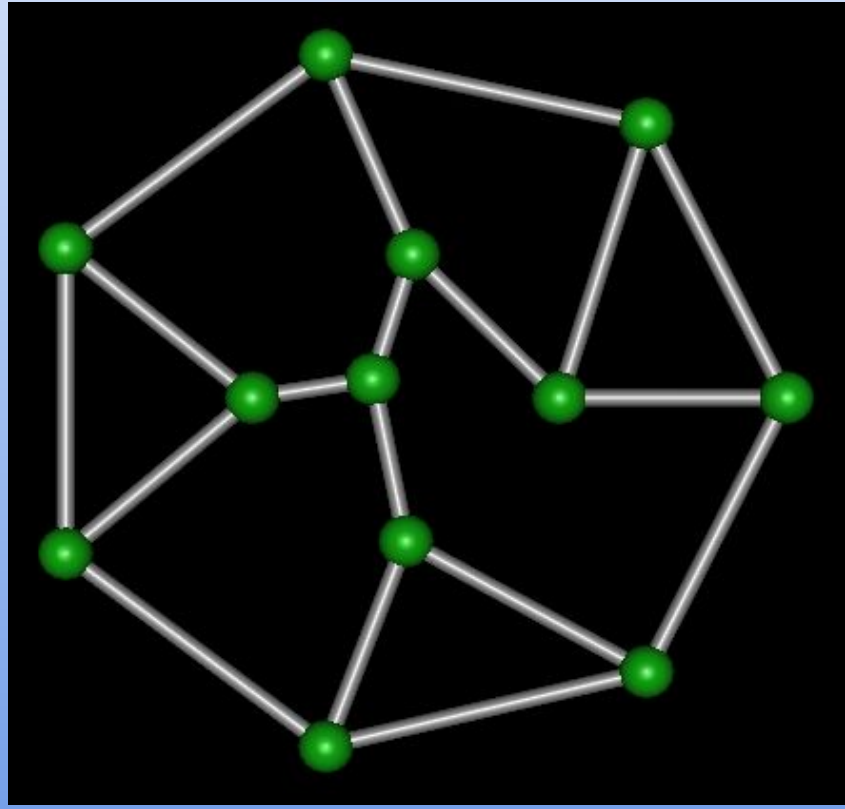
Vertex Cover u/ Graf Lengkap

- Graf simetris
- Jumlah tetangga semua verteks sama
- Vertek anggota VC dapat dipilih bebas
- Sisanya menjadi graf komplit dengan derajat lebih rendah
- → Anggota Vertex Cover adalah semua vertex dalam graf kurang satu verteks.



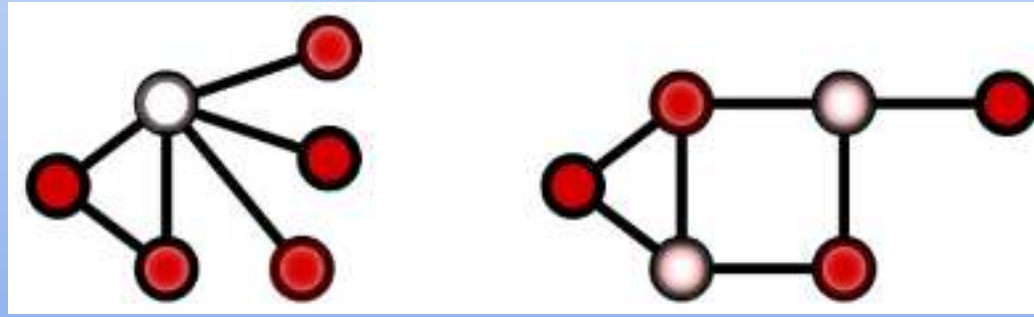
Vertex Cover

- Mudah mencari vertex cover minimal
- Tidak selalu betul untuk maksimum
- Graf ini juga mempunyai vertex dengan jumlah tetangga yang sama
- → Strategi preferensial derajat tinggi tidak bisa dilakukan
- Tetapi graf ini tidak simetris
  - Ada yang siklus 3 verteks
  - 4 verteks
  - 5 verteks
  - dan 6 verteks



Indipendent Set

- Diberikan graf G=(V, E)
- Cari himpunan terbesar IS $\subseteq$ V, agar setiap $\{u,v\} \subseteq$ IS, berlaku $(u,v) \notin$ E
- Beberapa IS saja (⇨)
- IS Terbesar (⇓)
- Relasi antara IS dan VC
  - IS $\cup$ VC = V dan IS $\cap$ VC = $\varnothing$
  - Bukti (dengan kontradiksi)



Problem Clique

- Diberikan graf G=(V, E)
- Cari himpunan terbesar CL $\subseteq$ V, dimana semua $\{u,v\} \subseteq$ CL, maka $(u,v) \in$ E
  - Jadi semacam subgraf lengkap didalam G
- Relasi antara IS dan Clique
  - X merupakan independent set dalam G iff verteks yang sama membentuk clique dalam graf komplemen-nya
  - Bukti (juga dengan kontradiksi)
    - Misalkan ada sebuah verteks termasuk IS di G tetapi bukan Clique dalam G^C
    - ...
    - Misalkan ada verteks termasuk Clique dalam G^C tetapi tidak termasuk independent set dalam G
    - ...

Problem yang Ekivalen?

- Solusi beberapa problems untuk bentuk general agak sulit ditemukan
- Tapi beberapa problems tersebut ternyata saling ekivalen
- i.e. Jika satu problem mempunyai solusi efisien, maka yang lain otomatis juga ada...