DESAIN DAN ANALISIS ALGORITMA PENYELESAIAN PERMASALAHAN PENUGASAN BERSYARAT DENGAN REPRESENTASI GRAF BIPARTIT

Penyusun Tugas Akhir:

Muhammad Izzuddin

NRP. 5112 100 012

Dosen Pembimbing:

Arya Yudhi Wijaya, S.Kom., M.Kom.

NIP. 19840904 201012 1 002

Rully Soelaiman, S.Kom., M.Kom.

NIP. 19700213 199402 1 001

1. PENDAHULUAN
2. TINJAUAN PUSTAKA
3. STRATEGI PENYELESAIAN
4. UJI COBA & EVALUASI
5. KESIMPULAN

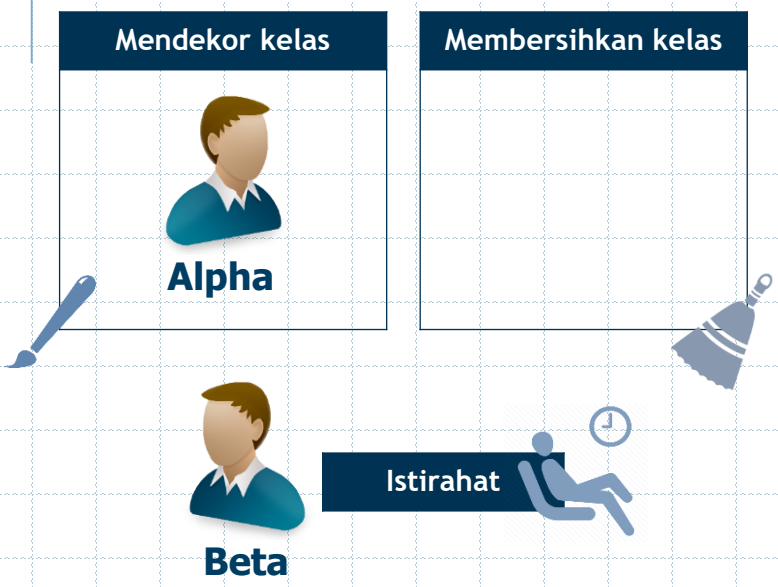
LATAR BELAKANG

- Dalam teori graf, permasalahan penugasan bersyarat membahas bagaimana menemukan susunan pemberian tugas atau pekerjaan pada pekerja agar sebuah pekerjaan tersebut dapat dilakukan dengan seoptimal mingkin dengan beberapa konstrain permasalahan yang ada.
- Permasalahan bersyarat yang diangkat dalam tugas akhir ini diambil dari persoalan SPOJ Klasik 6819 yang berjudul “Yet Another Assignment Problem” dengan kode soal ASSIGN5.

ILUSTRASI PERSOALAN
YET ANOTHER ASSIGNMENT PROBLEM

- Untuk mendekorasi sebuah kelas, Alpha harus mengerjakan hal tersebut selama 2 menit ditambah Beta selama 5 menit dan untuk membersihkan kelas, Alpha harus bekerja selama 5 menit ditambah Beta selama 1 menit.

Menit 1

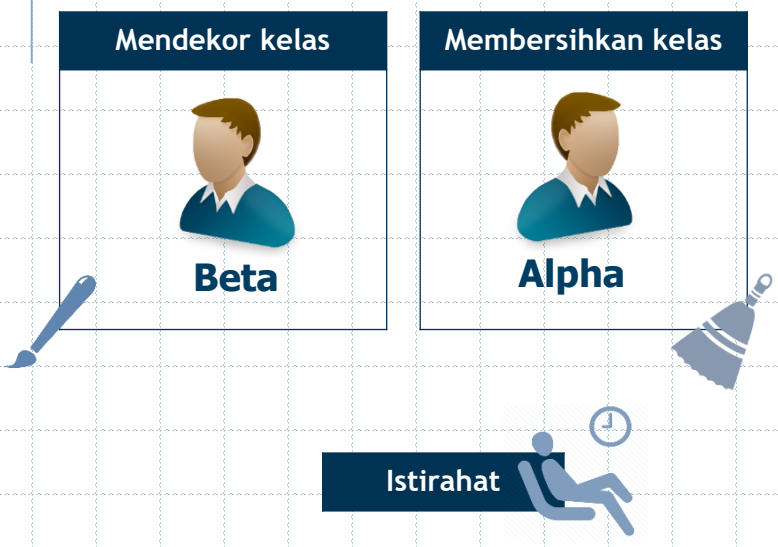


Menit	Alpha	Beta
1	Mendekor kelas	Istirahat

ILUSTRASI PERSOALAN
YET ANOTHER ASSIGNMENT PROBLEM

- Untuk mendekorasi sebuah kelas, Alpha harus mengerjakan hal tersebut selama 2 menit ditambah Beta selama 5 menit dan untuk membersihkan kelas, Alpha harus bekerja selama 5 menit ditambah Beta selama 1 menit.

Menit 2

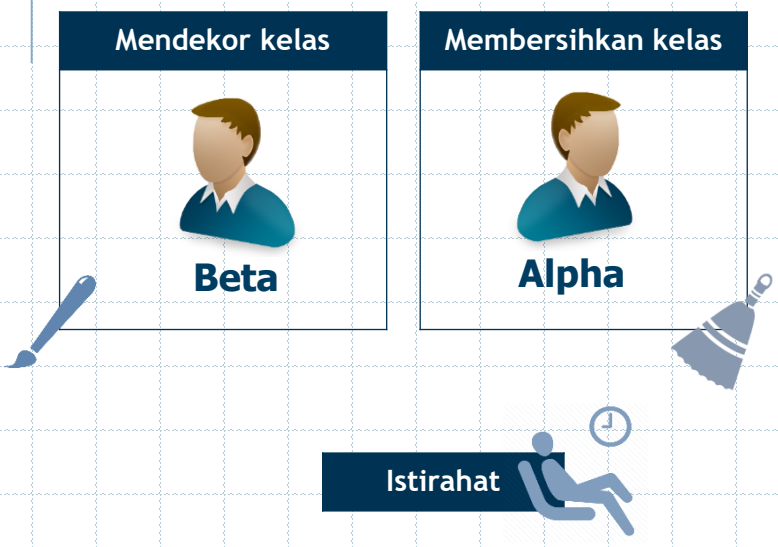


Menit	Alpha	Beta
1	Mendekor kelas	Istirahat
2	Membersihkan kelas	Mendekor kelas

ILUSTRASI PERSOALAN
YET ANOTHER ASSIGNMENT PROBLEM

- Untuk mendekorasi sebuah kelas, Alpha harus mengerjakan hal tersebut selama 2 menit ditambah Beta selama 5 menit dan untuk membersihkan kelas, Alpha harus bekerja selama 5 menit ditambah Beta selama 1 menit.

Menit 3

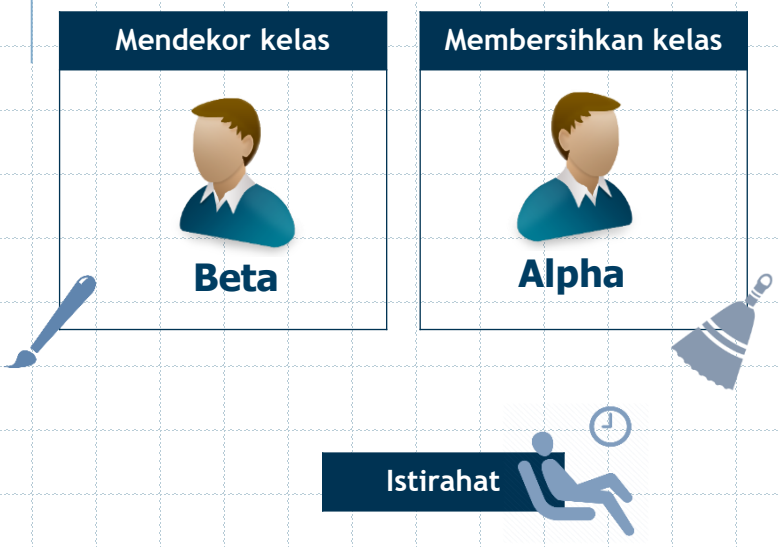


Menit	Alpha	Beta
1	Mendekor kelas	Istirahat
2	Membersihkan kelas	Mendekor kelas
3	Membersihkan kelas	Mendekor kelas

ILUSTRASI PERSOALAN
YET ANOTHER ASSIGNMENT PROBLEM

- Untuk mendekorasi sebuah kelas, Alpha harus mengerjakan hal tersebut selama 2 menit ditambah Beta selama 5 menit dan untuk membersihkan kelas, Alpha harus bekerja selama 5 menit ditambah Beta selama 1 menit.

Menit 4

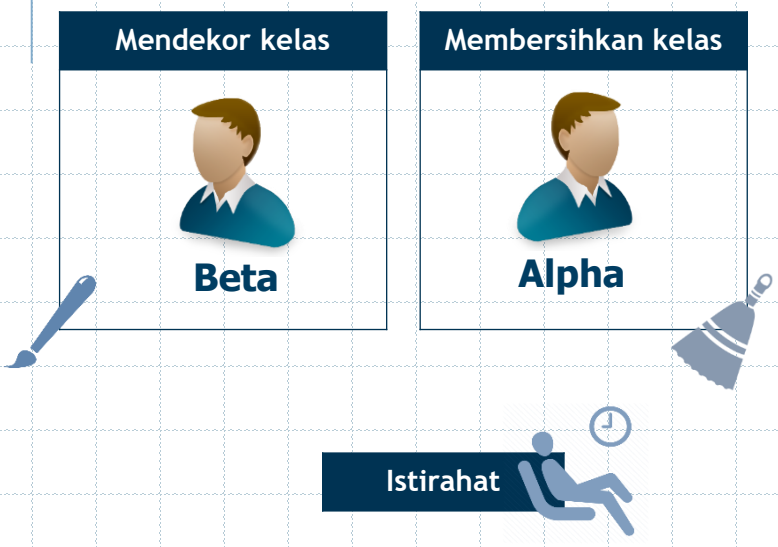


Menit	Alpha	Beta
1	Mendekor kelas	Istirahat
2	Membersihkan kelas	Mendekor kelas
3	Membersihkan kelas	Mendekor kelas
4	Membersihkan kelas	Mendekor kelas

ILUSTRASI PERSOALAN
YET ANOTHER ASSIGNMENT PROBLEM

- Untuk mendekorasi sebuah kelas, Alpha harus mengerjakan hal tersebut selama 2 menit ditambah Beta selama 5 menit dan untuk membersihkan kelas, Alpha harus bekerja selama 5 menit ditambah Beta selama 1 menit.

Menit 5

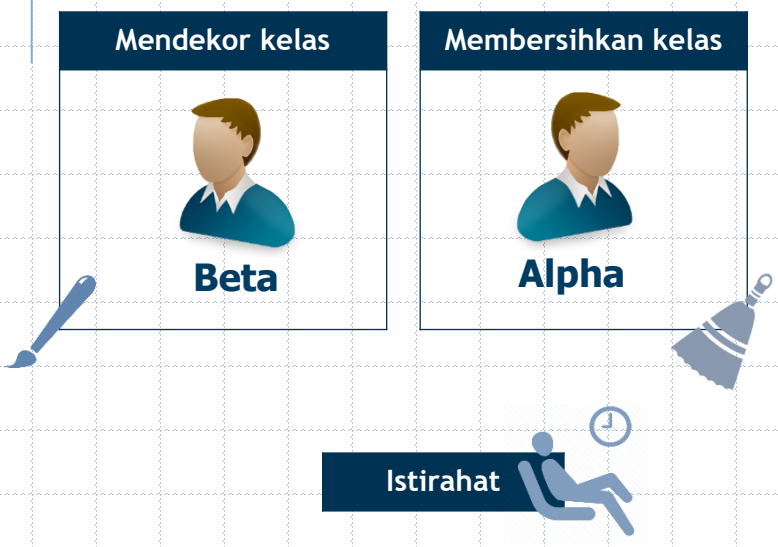


Menit	Alpha	Beta
1	Mendekor kelas	Istirahat
2	Membersihkan kelas	Mendekor kelas
3	Membersihkan kelas	Mendekor kelas
4	Membersihkan kelas	Mendekor kelas
5	Membersihkan kelas	Mendekor kelas

ILUSTRASI PERSOALAN
YET ANOTHER ASSIGNMENT PROBLEM

- Untuk mendekorasi sebuah kelas, Alpha harus mengerjakan hal tersebut selama 2 menit ditambah Beta selama 5 menit dan untuk membersihkan kelas, Alpha harus bekerja selama 5 menit ditambah Beta selama 1 menit.

Menit 6

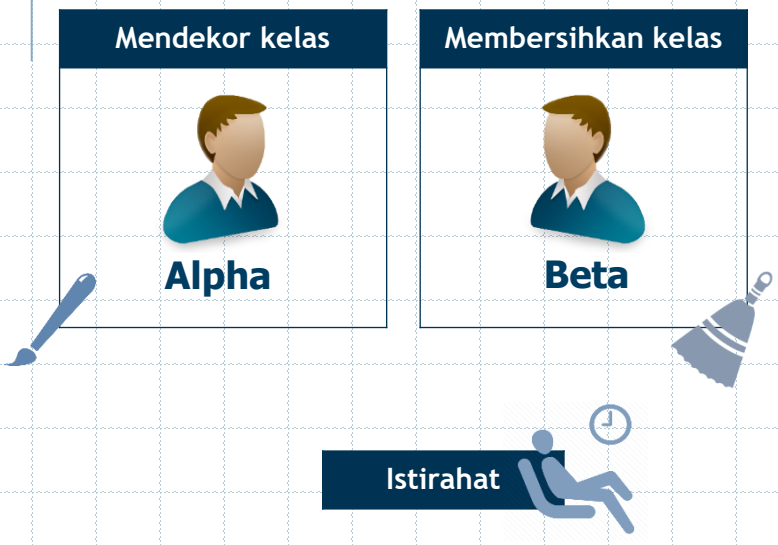


Menit	Alpha	Beta
1	Mendekor kelas	Istirahat
2	Membersihkan kelas	Mendekor kelas
3	Membersihkan kelas	Mendekor kelas
4	Membersihkan kelas	Mendekor kelas
5	Membersihkan kelas	Mendekor kelas
6	Membersihkan kelas	Mendekor kelas

ILUSTRASI PERSOALAN
YET ANOTHER ASSIGNMENT PROBLEM

- Untuk mendekorasi sebuah kelas, Alpha harus mengerjakan hal tersebut selama 2 menit ditambah Beta selama 5 menit dan untuk membersihkan kelas, Alpha harus bekerja selama 5 menit ditambah Beta selama 1 menit.

Menit 7



Menit	Alpha	Beta
1	Mendekor kelas	Istirahat
2	Membersihkan kelas	Mendekor kelas
3	Membersihkan kelas	Mendekor kelas
4	Membersihkan kelas	Mendekor kelas
5	Membersihkan kelas	Mendekor kelas
6	Membersihkan kelas	Mendekor kelas
7	Mendekor kelas	Membersihkan kelas

RUMUSAN MASALAH

- Bagaimana menyelesaikan permasalahan penugasan bersyarat pada SPOJ Klasik 6819 **Yet Another Assignment Problem** (ASSIGN5) dengan representasi graf bipartit?
- Bagaimana mengkonstruksi dan membuktikan kebenaran algoritma Hopcroft-Karp dengan menggunakan permasalahan SPOJ Klasik 19295 **The Dilemma of Idli**?
- Bagaimana hasil efisiensi waktu penyelesaian permasalahan penugasan bersyarat pada SPOJ Klasik 6819 **Yet Another Assignment Problem** (ASSIGN5) dengan representasi graf bipartit?

BATASAN MASALAH

- Implementasi algoritma menggunakan bahasa pemrograman C++.
- Batasan permasalahan pembuktian Algoritma Hopcroft-Karp yang digunakan sesuai pada permasalahan SPOJ “The Dilemma of Idli” dengan kode soal WPC5G.
- Batasan permasalahan yang digunakan sesuai pada permasalahan SPOJ “Yet Another Assignment Problem” dengan kode soal ASSIGN5.

TUJUAN

- Melakukan desain serta analisis penyelesaian permasalahan penugasan bersyarat pada SPOJ Klasik 6819 Yet Another Assignment Problem (ASSIGN5) dengan representasi graf bipartit.
- Menganalisis hasil efisiensi waktu penyelesaian permasalahan penugasan bersyarat pada SPOJ Klasik 6819 Yet Another Assignment Problem (ASSIGN5) dengan representasi graf bipartit.

1. PENDAHULUAN
2. TINJAUAN PUSTAKA
3. STRATEGI PENYELESAIAN
4. UJI COBA & EVALUASI
5. KESIMPULAN

SPOJ *YET ANOTHER ASSIGNMENT PROBLEM*

- Terdapat beberapa pekerjaan dan beberapa orang yang akan melakukan pekerjaan tersebut.
- Setiap pekerjaan harus dilakukan oleh semua orang yang ada serta setiap orang memiliki waktu yang dibutuhkan tersendiri dalam menyelesaikan pekerjaan tersebut.
- Setiap orang hanya dapat mengerjakan sebuah pekerjaan dan sebuah pekerjaan hanya dapat dikerjakan oleh satu orang dalam satu satuan waktu.
- Pekerjaan yang dimaksud juga bersifat independen dalam artian dapat dilakukan dan dihentikan kapanpun oleh setiap orang.

SPOJ *YET ANOTHER ASSIGNMENT PROBLEM*

- Diberikan m pekerjaan dan n pekerja, serta m baris kelompok bilangan. Setiap baris i berisi n buah bilangan dimana bilangan ke j-nya adalah A_{ij} yang berarti teman ke-j harus melakukan pekerjaan i selama A_{ij} menit

- Diminta untuk mencari waktu minimal yang diperlukan untuk melakukan semua pekerjaan tersebut beserta susunan penugasan pada menit ke-0 yang mungkin terjadi.

BATASAN PERMASALAHAN
YET ANOTHER ASSIGNMENT PROBLEM

- Batasan dalam permasalahan:
  - Jumlah pekerjaan m : 1 ≤ m ≤ 2.000
  - Jumlah pekerja n : 1 ≤ n ≤ 2.000
  - Lama pengerjaan sebuah pekerjaan oleh seseorang A_{ij} : $0 \leq A_{ij} \leq 10^6$
- Waktu eksekusi maksimal 0.132 - 0.661 detik
- Penggunaan memori maksimal 1536 MB

1. PENDAHULUAN
2. TINJAUAN PUSTAKA
3. STRATEGI PENYELESAIAN
4. UJI COBA & EVALUASI
5. KESIMPULAN

STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Terdapat 3 tahapan penyelesaian permasalahan *Yet Another Assignment Problem*

1. Menentukan waktu minimum penyelesaian semua pekerjaan (α).
2. Melakukan proses expanding matrix
3. Mencari *maximum bipartite matching*

Sebelum mengkonstruksi solusi dari permasalahan ini, terlebih dahulu dibuktikan kebenaran algoritma Hopcroft-Karp yang digunakan untuk mencari *maximum bipartite matching* dengan menggunakan permasalahan *The Dilemma of Idli*

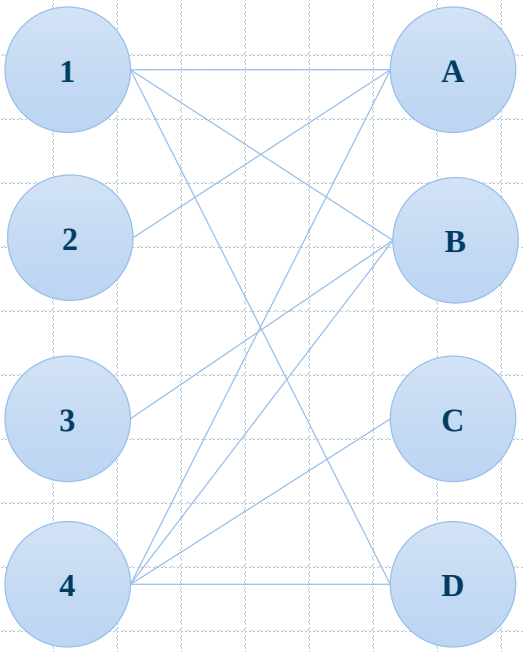
ALGORITMA *HOPCROFT-KARP*

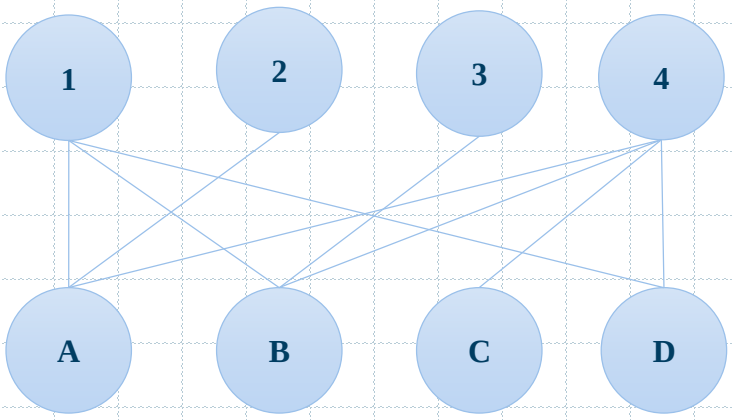
- Algoritma *Hopcroft-Karp* adalah algoritma yang memerlukan *bipartite* graf sebagai masukannya dan menghasilkan *maximum-size matching* didalam graf tersebut sebagai keluarannya dengan kompleksitas $O(|V|\sqrt{|E|})$.
- Dimisalkan terdapat sebuah graf $G = (X \cup Y, E)$ dan M adalah matching yang di dapat dari graf tersebut. Secara garis besar, algoritma ini berjalan sebagai berikut:
  1. Sebuah *breadth-first search* digunakan untuk membagi *vertex-vertex* menjadi beberapa *layer*. Sebuah *layer* dalam hal ini diartikan sebagai sebuah *level* dalam *tree*, dan setiap *level* dalam *tree* menunjukkan panjang augmented path yang dapat di bentuk.

ALGORITMA *HOPCROFT-KARP*

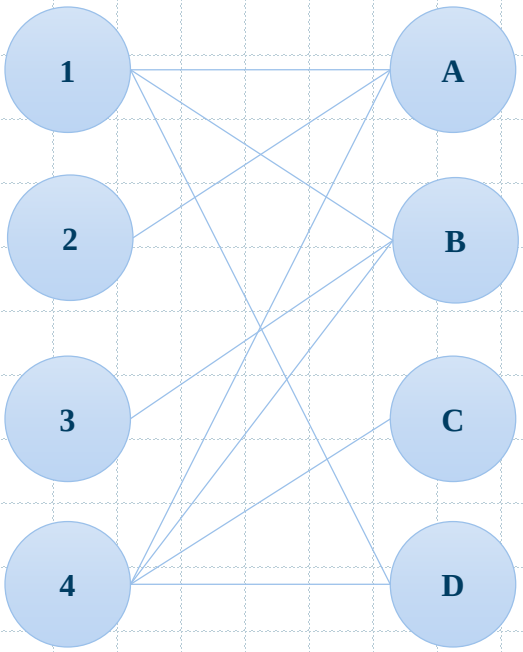
2. Pencarian dimulai melalui *unmatched vertex* pada X. Pencarian path dilakukan dengan men-*treverse unmatched edge dan matched edge* secara bergantian.
3. Pencarian akan berhenti pada sebuah layer-k dimana terdapat satu atau beberapa *unmatched vertex* pada Y tercapai.
4. Setelah layer pada graf tersebut terbentuk, melalui *depth-first search,* sebuah himpunan maksimal dari *augmenting path* terpendek yang memiliki panjang k dapat dicari dengan awal *unmatched vertex* pada X dan berakhir pada *unmatched vertex* Y.
5. Untuk setiap *augmenting path* yang ditemukan, semua *unmatched vertex* tersebut diubah menjadi *matched vertex* begitu pula sebaliknya.

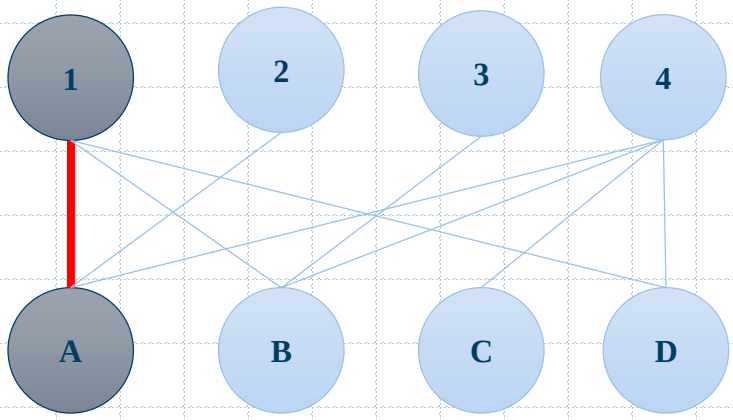
ALGORITMA *HOPCROFT-KARP*





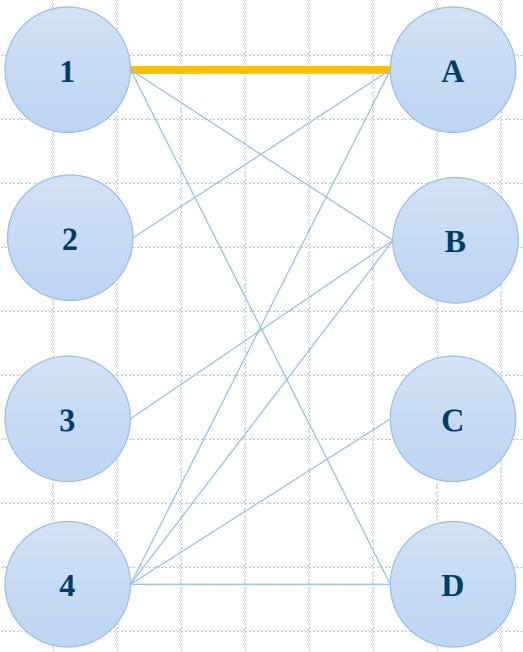
ALGORITMA *HOPCROFT-KARP*

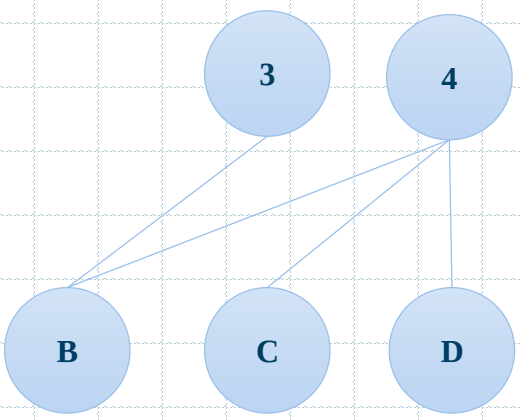




$m = \{(1, A)\}$

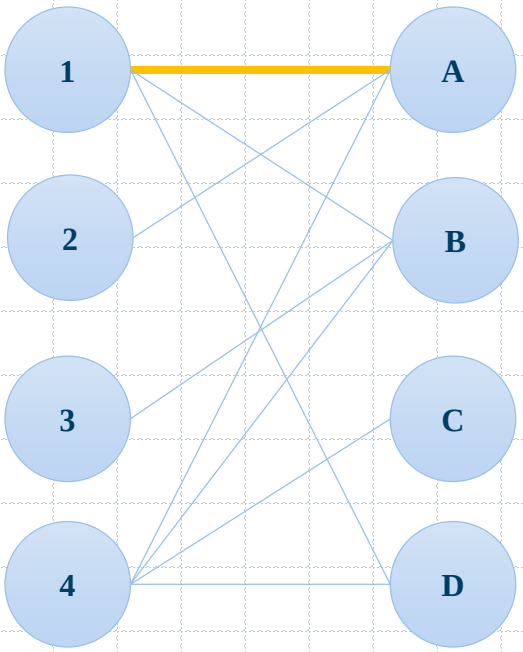
ALGORITMA *HOPCROFT-KARP*

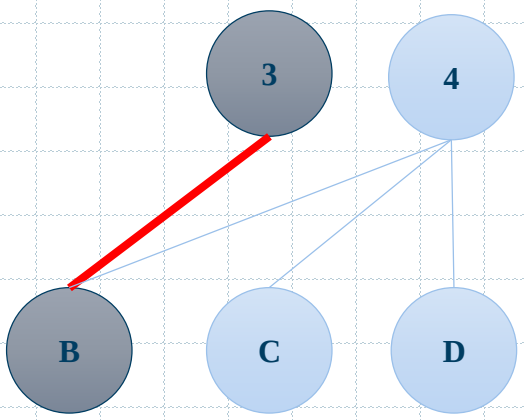




$m = \{(1, A)\}$

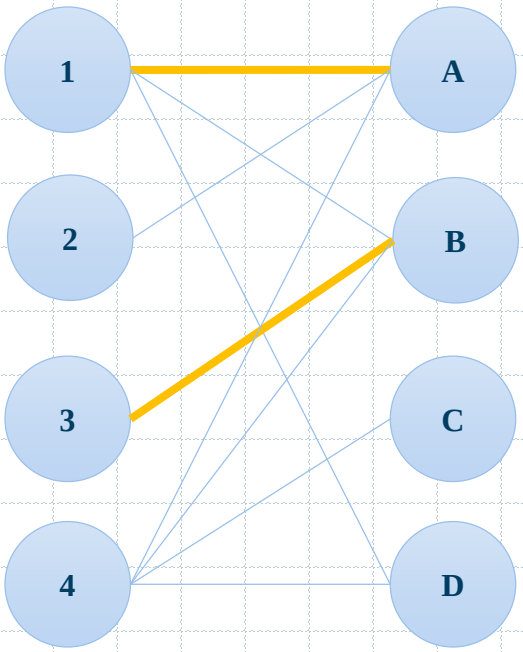
ALGORITMA *HOPCROFT-KARP*

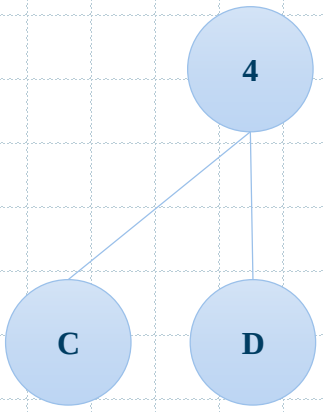




$$m = \{(1, A), (3, B)\}$$

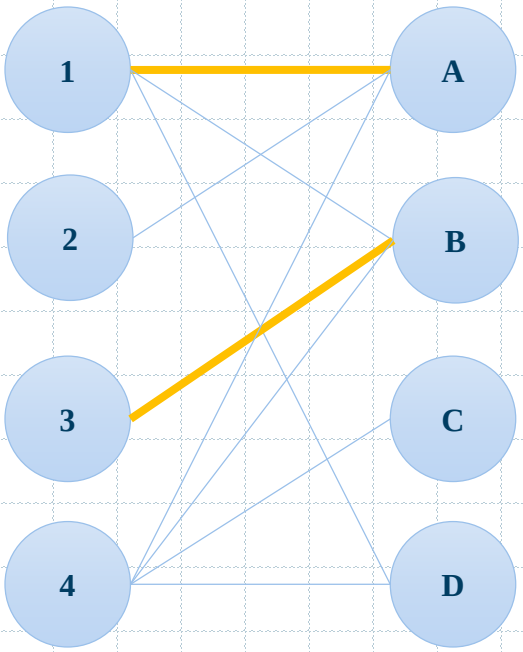
ALGORITMA *HOPCROFT-KARP*

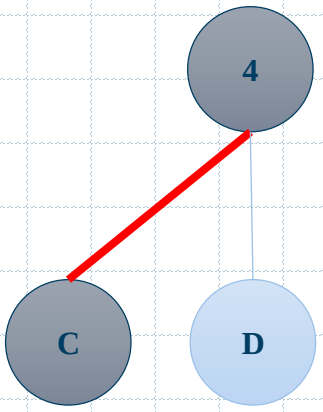




$$m = \{(1, A), (3, B)\}$$

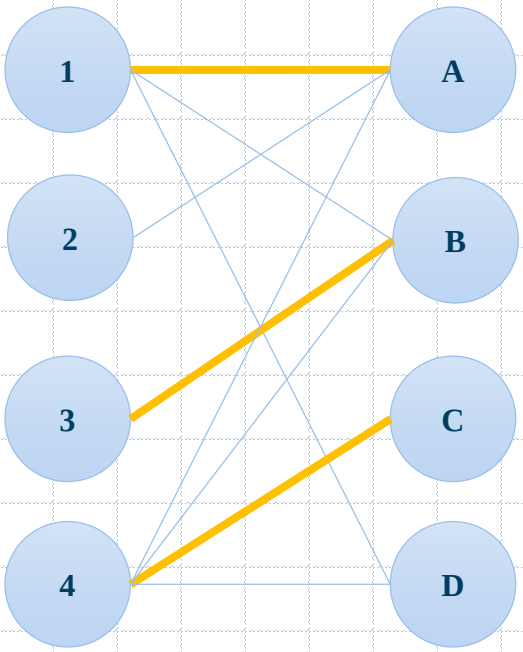
ALGORITMA *HOPCROFT-KARP*





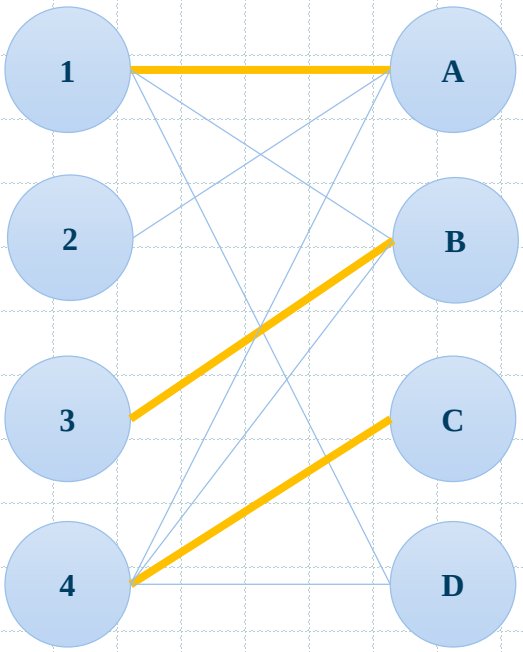
$$m = \{(1, A), (3, B), (4, C)\}$$

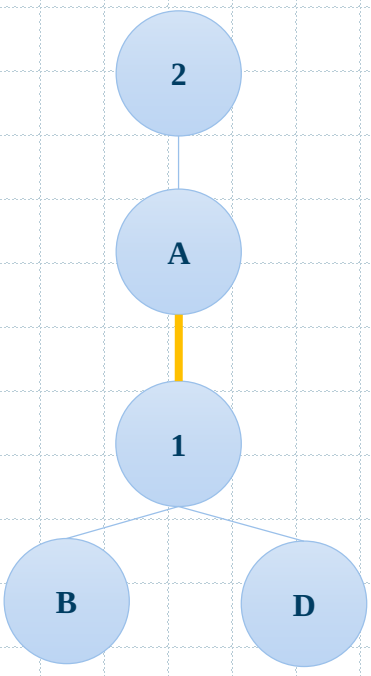
ALGORITMA *HOPCROFT-KARP*



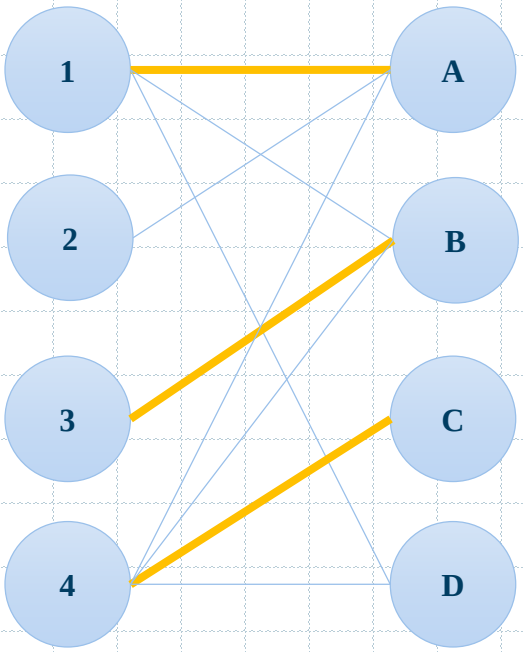
$$m = \{(1, A), (3, B), (4, C)\}$$

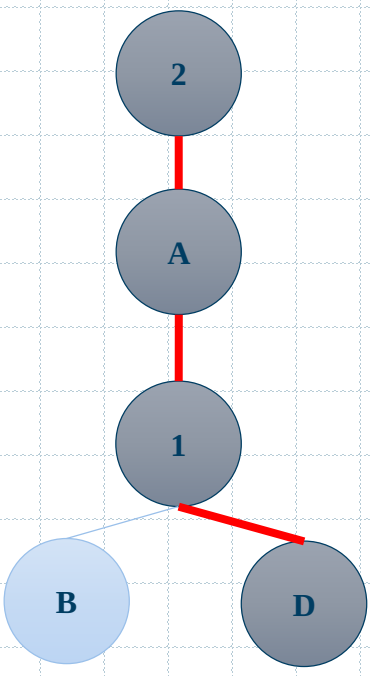
ALGORITMA *HOPCROFT-KARP*





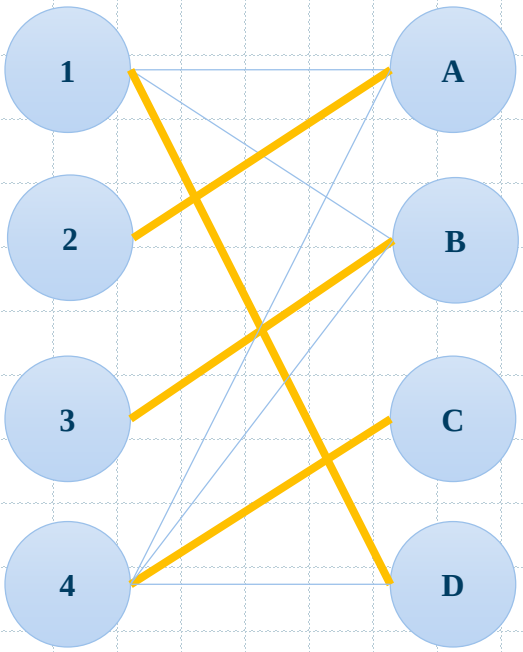
ALGORITMA *HOPCROFT-KARP*





$$m = \{(2, A), (1, D)\}$$

ALGORITMA *HOPCROFT-KARP*



$$m = \{(2, A), (1, D)\}$$

SPOJ *THE DILEMMA OF IDLI*

- Terdapat sebuah negeri yang penduduknya tebagi menjadi 2 bagian yaitu penduduk A dan B.
- Penduduk A menyukai beberapa orang tertentu dari grup *Committee of the Galactical Wars*, dan membenci beberapa orang tertentu dari grup *Inter-Galactic Parliament*. Penduduk B merupakan kebalikan sifat dari penduduk B.
- Beberapa orang dari *Committee of the Galactical Wars* serta *Inter-Galactic Parliament harus dicurigai* agar masyarakatnya merasa tenang dan puas.
- Seorang penduduk dapat dikatakan puas jika dan hanya jika semua orang yang dia suka, tidak dicurigai dan semua orang yang dia benci, dicurigai oleh Idli.

SPOJ *THE DILEMMA OF IDLI*

- Diberikan n orang penduduk. Setiap penduduk dideskripsikan tergolong dalam kelompok penduduk A atau B serta memiliki daftar orang yang dia suka dan daftar orang yang dibenci.

- Diminta untuk mencari sebuah nilai yang menunjukkan berapa banyak penduduk yang dapat dipuaskan oleh Idli.

BATASAN PERMASALAHAN
THE DILEMMA OF IDLI

- Batasan dalam permasalahan:
  - $1 \leq n1 \leq 1000000$
  - $1 \leq n2 \leq 1000000$
  - $1 \leq n \leq 500$
  - $0 \leq$ banyaknya anggota *CoGW* / *IGP* yang disukai oleh seorang penduduk ≤ 50.
  - $0 \leq$ banyaknya anggota *CoGW* / *IGP* yang dibenci oleh seorang penduduk ≤ 50.
- Waktu eksekusi maksimal 1 detik
- Penggunaan memori maksimal 1536 MB

STRATEGI PENYELESAIAN THE DILEMMA OF IDLI

- Pada deskripsi soal tersebut diketahui bahwa **seorang penduduk dikatakan puas jika dan hanya jika semua orang yang dia suka, tidak dicurigai dan semua orang yang dia benci, dicurigai oleh Idli.**
- Dengan menegasikan premis tersebut didapatkan bahwa seorang penduduk dikatakan tidak puas jika terdapat salah satu orang yang dia suka dicurigai atau terdapat salah satu orang yang dia benci, tidak dicurigai.
- Dimisalkan memilih memuaskan seorang penduduk A, sebagai konsekuensinya, semua penduduk B yang menyukai orang yang dibenci oleh A merasa tidak puas, begitu juga sebaliknya.

STRATEGI PENYELESAIAN THE DILEMMA OF IDLI

- Permasalahan ini dapat direpresentasikan dengan *Bipartite Graph* $G = (A \cup B, E)$ dimana A adalah setiap orang dari penduduk A merupakan dan B adalah setiap orang dari penduduk B. dan E merupakan representasi dari hubungan setiap orang yang dibenci penduduk A_i tetapi disukai penduduk B_i ataupun sebaliknya.
- *Maximum Bipartite Matching* pada graf tersebut akan mengembalikan nilai t yaitu total *vertex* minimum pada A dan B yang dapat diambil dari hubungan *edge* yang ada atau dengan kata lain banyaknya orang yang tidak dapat dipuaskan.

$$\boldsymbol{Max\ Penduduk\ Puas = jumlah\ penduduk - t}$$

UJI COBA KEBENARAN THE DILEMMA OF IDLI

- Melalui permasalahan ***The Dilemma of Idli***, *implementasi algoritma Hopcroft-Karp yang dilakukan mendapat umpan balik* ***accepted*** dari situs penilaian daring SPOJ.

17068371	2016-06-08 06:36:43	The dilemma of Idli	**accepted**	0.31	26M	C++ 4.3.2

- Waktu yang tercepat yang dibutuhkan program adalah 0.31 detik dan memori yang dibutuhkan adalah 26 MB.

STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

1. **Menentukan waktu minimum penyelesaian semua pekerjaan (α).**

- Waktu minimum yang diperlukan untuk melakukan semua pekerjaan yang ada adalah jumlah waktu terbanyak yang dilakukan seseorang untuk menyelesaikan semua pekerjaannya atau waktu terbanyak yang dibutuhkan oleh sebuah pekerjaan agar diselesaikan oleh semua orang. Misal waktu minimum yang dibutuhkan untuk menyelesaikan semua pekerjaan adalah α, maka:

$$\alpha = Max\left(\sum_{i=0,j=0}^{m-1,n-1} A_{i,j}\,,\ \sum_{i=0,j=0}^{n-1,m-1} A_{j,i}\right)$$

	Pekerja 1	Pekerja 2	Pekerja 3
Pekerjaan 1	2	2	4
Pekerjaan 2	1	2	0

Ilustrasi Permasalahan

- Berdasarkan ilustrasi permaslahan tersebut, didapatkan nilai $\alpha = 8$

STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

2. **Melakukan proses expanding matrix**

- Dibentuk sebuah *matrix* awal yang berukuran $m * n$ sesuai dengan permasalahan.
- Dibentuk sebuah *matrix* dengan ukuran $(n + m) * (m + n)$ dan mengisi $m * n$ *matrix* pertama dalam *matrix* baru tersebut sesuai dengan *matrix* awal.
- mengisi semua baris dan kolom yang belum terisi dengan sebuah nilai sedemikian rupa sehingga setiap baris dan kolom memiliki jumlah tepat α.

2	2	4
1	2	0

(a)

2	2	4	0	0
1	2	0	0	5
5	0	0	3	0
0	4	0	4	0
0	0	4	1	3

(b)

STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

3. Mencari *maximum bipartite matching*

Secara umum algoritma yang digunakan untuk menyelesaikan permasalahan ini berjalan sebagai berikut :

1. Membentuk *bipartite* graf $G = (X \cup Y, E)$ dengan X merupakan representasi dari kolom (pekerja) dan Y merupakan representasi dari baris (pekerjaan) dari *matrix* yang baru. Sebuah *edge* pada E menghubungkan *vertex* pada X dan Y apabila nilai $A_{i,j}$ pada matrix tersebut tidak 0.

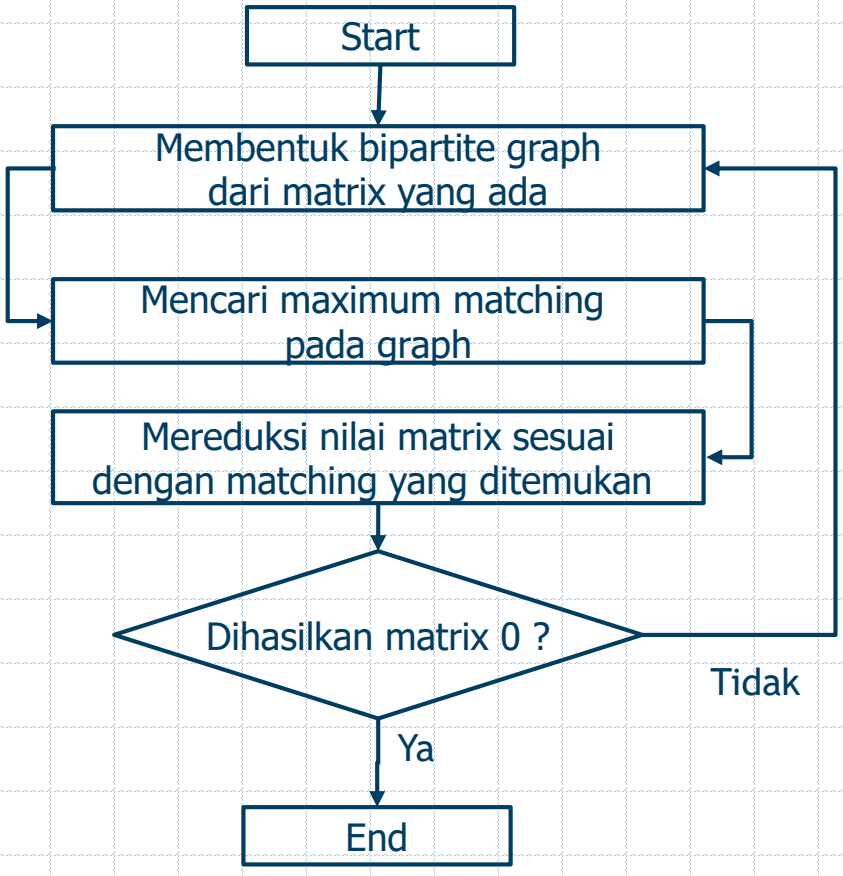
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

3. Mencari *maximum bipartite matching*

2. Mencari *maximum matching* yang terdapat pada graf tersebut. Apabila sebuah kolom i memiliki pasangan dengan baris j, artinya pekerja ke-i mengerjakan pekerjaan ke-j dalam suatu waktu tersebut. Apabila $j > m$ maka pekerja tersebut tidak melakukan pekerjaan apapun di satuan waktu tertentu.
3. Setiap baris j dan kolom i yang berpasangan, kurangi nilai $A_{i,j}$ pada matrix tersebut sebanyak 1.
4. Ulangi setiap proses 1 – 3 hingga matrix tersebut menjadi *matrix* dengan semua isinya adalah 0.

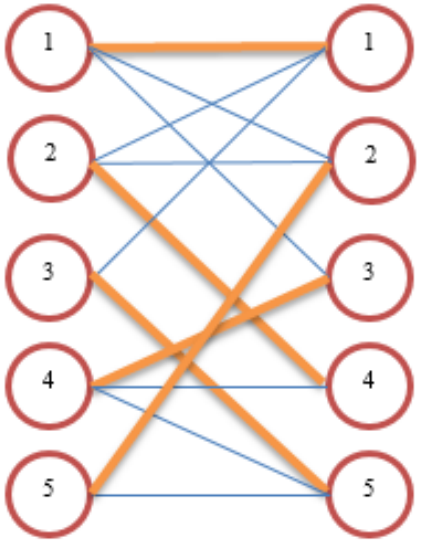
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 1

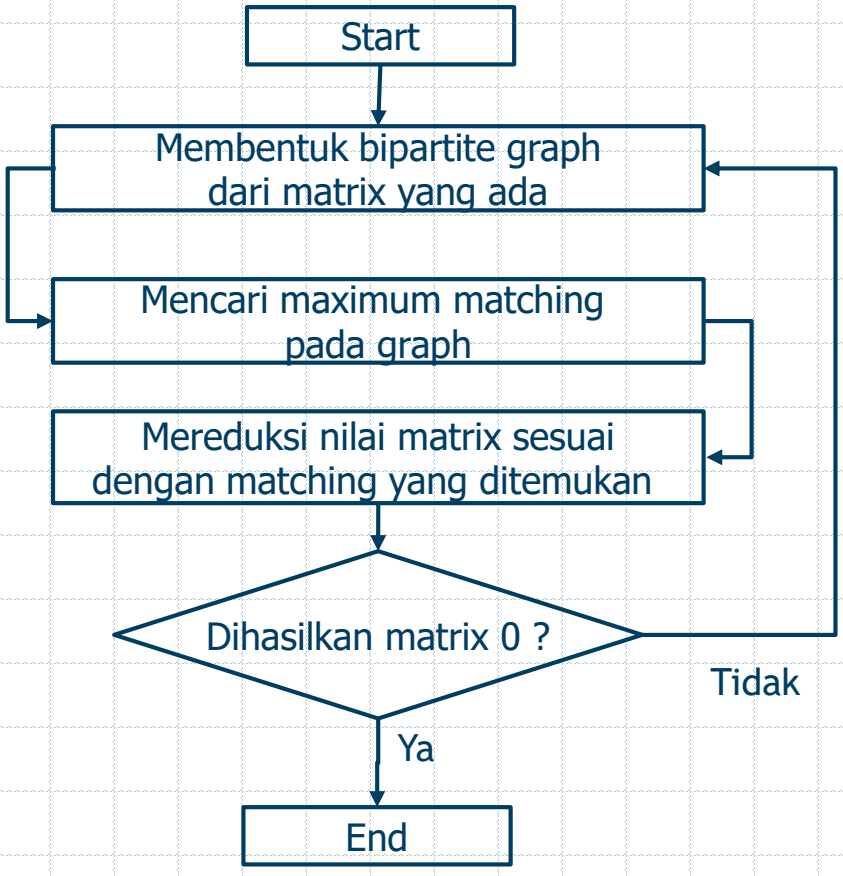
1	2	4	0	0
1	2	0	0	4
5	0	0	2	0
0	3	0	4	0
0	0	3	1	3



$m = \{(1,1), (2,4), (3,5), (4,3), (5,2)\}.$

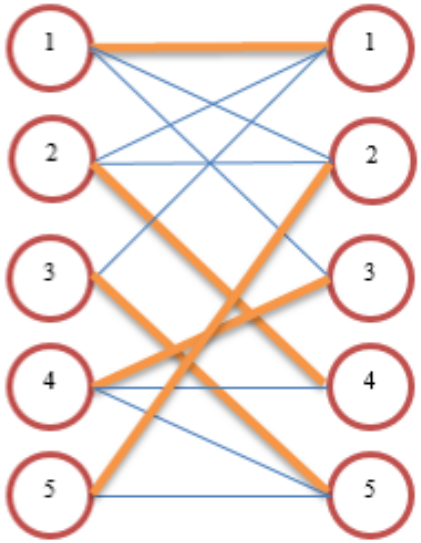
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 2

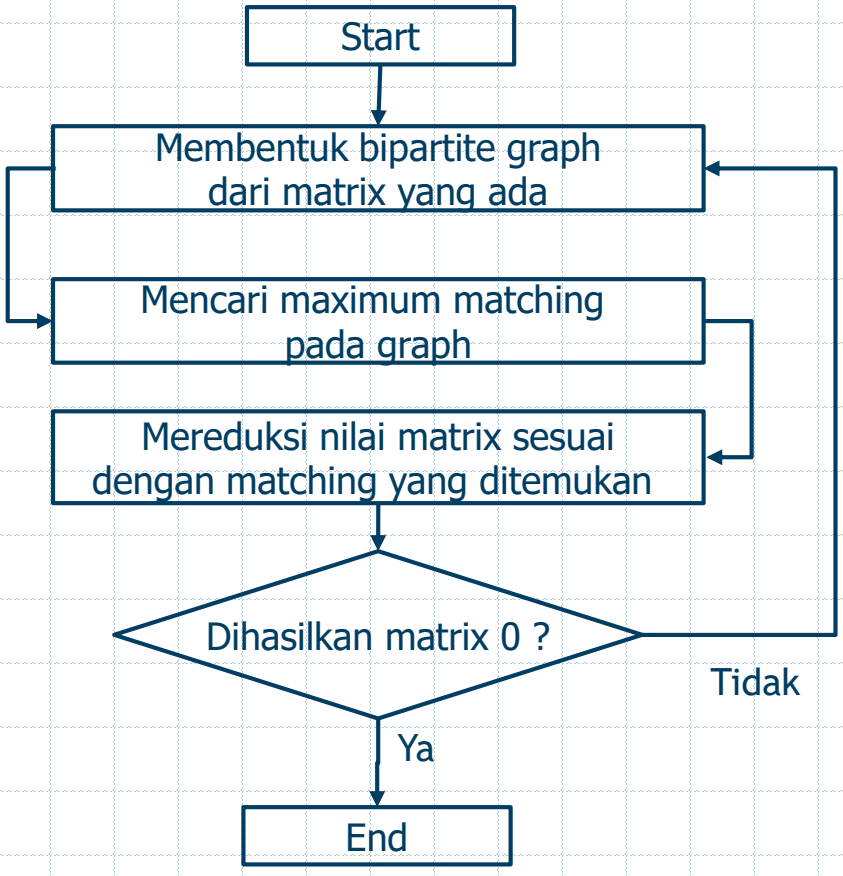
0	2	4	0	0
1	2	0	0	3
5	0	0	1	0
0	2	0	4	0
0	0	2	1	3



$m = \{(1,1), (2,4), (3,5), (4,3), (5,2)\}.$

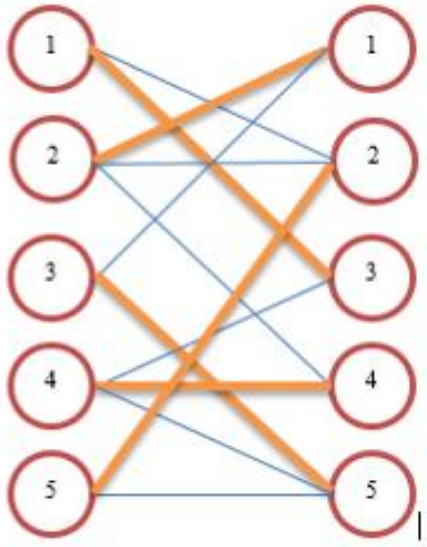
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 3

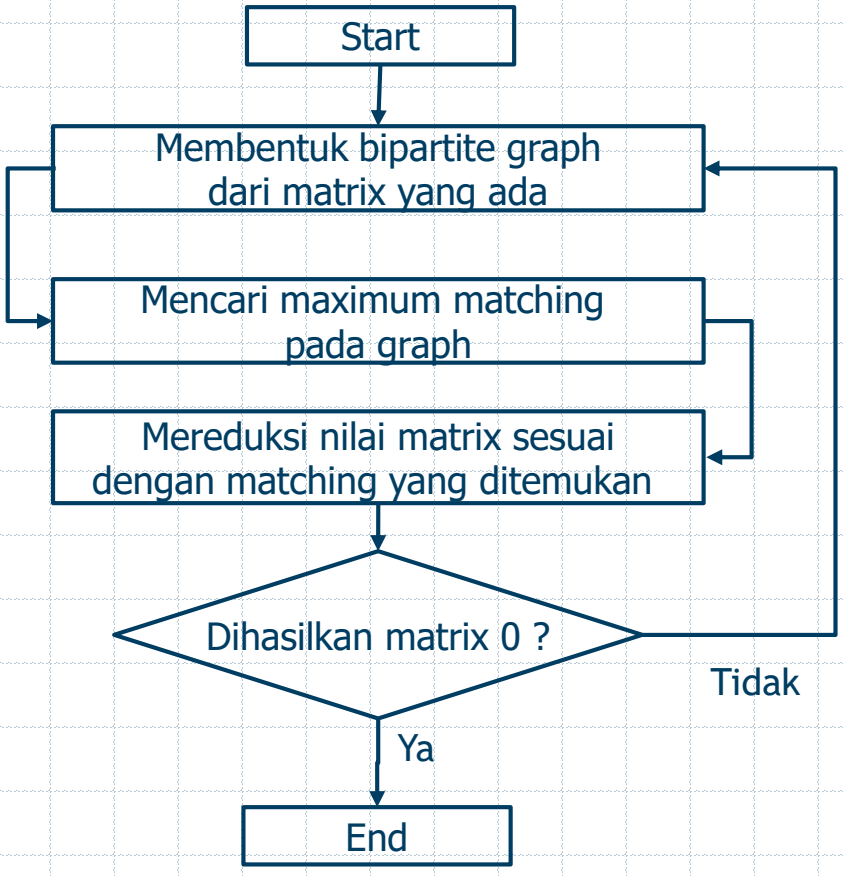
0	1	4	0	0
1	2	0	0	2
4	0	0	1	0
0	2	0	3	0
0	0	1	1	3



$m = \{(1,3), (2,1), (3,5), (4,4), (5,2)\}.$

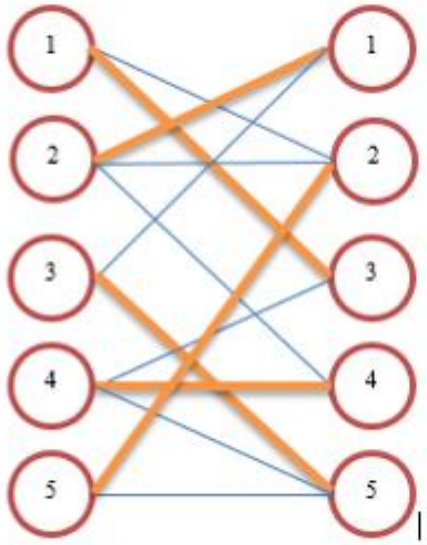
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 4

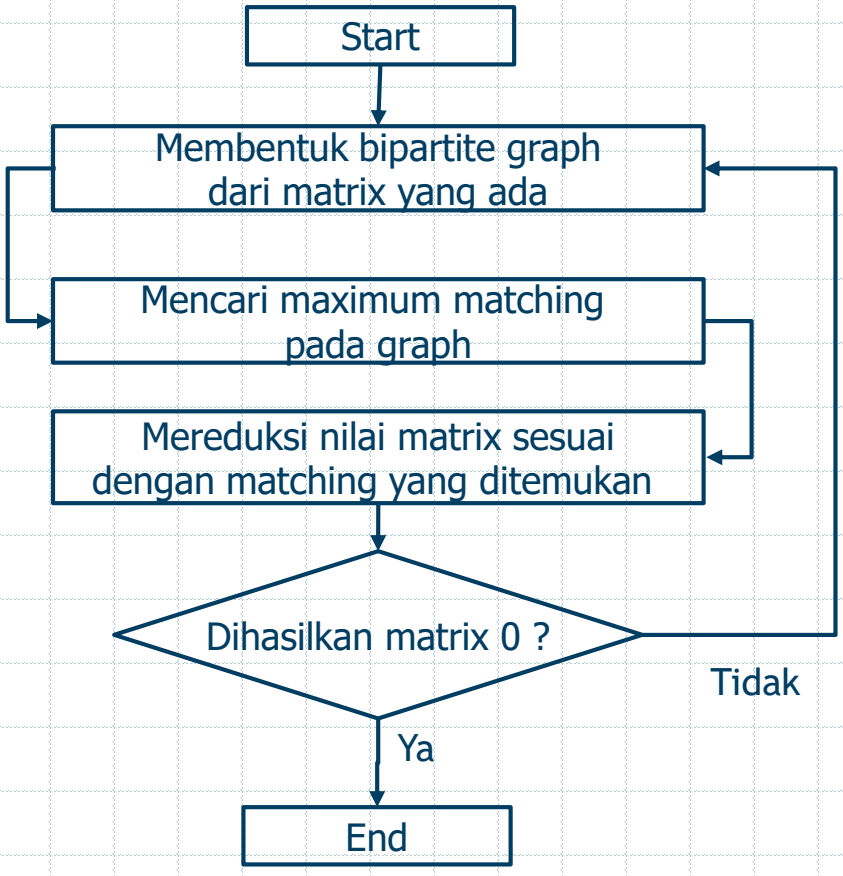
0	0	4	0	0
1	2	0	0	1
3	0	0	1	0
0	2	0	2	0
0	0	0	1	3



$m = \{(1,3), (2,1), (3,5), (4,4), (5,2)\}.$

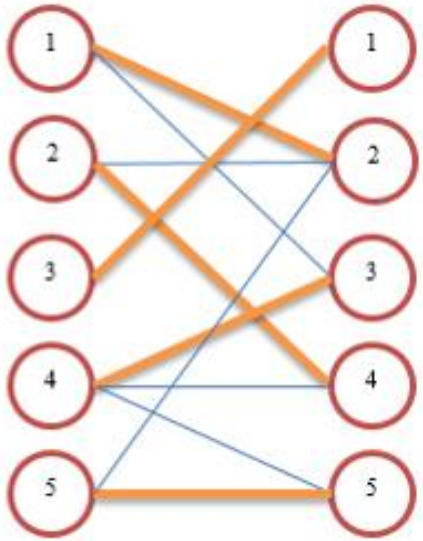
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 5

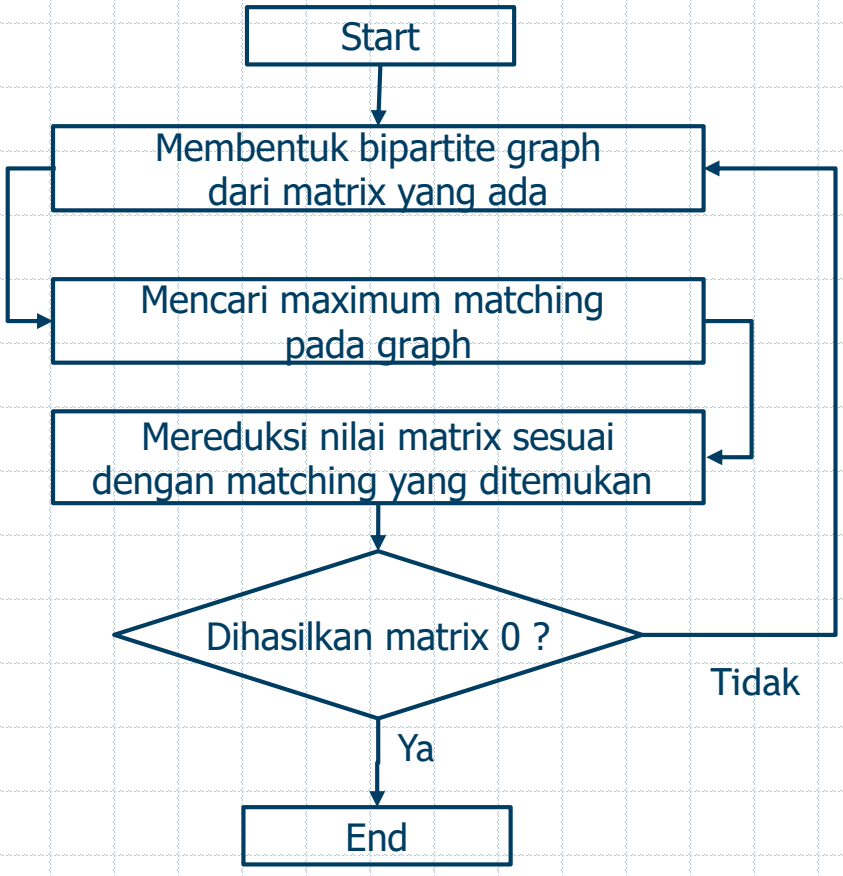
0	0	3	0	0
0	2	0	0	1
3	0	0	0	0
0	1	0	2	0
0	0	0	1	2



$m = \{(1,2), (2,4), (3,1), (4,3), (5,5)\}.$

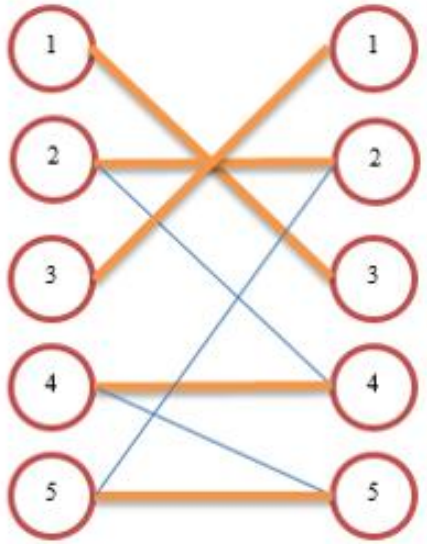
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 6

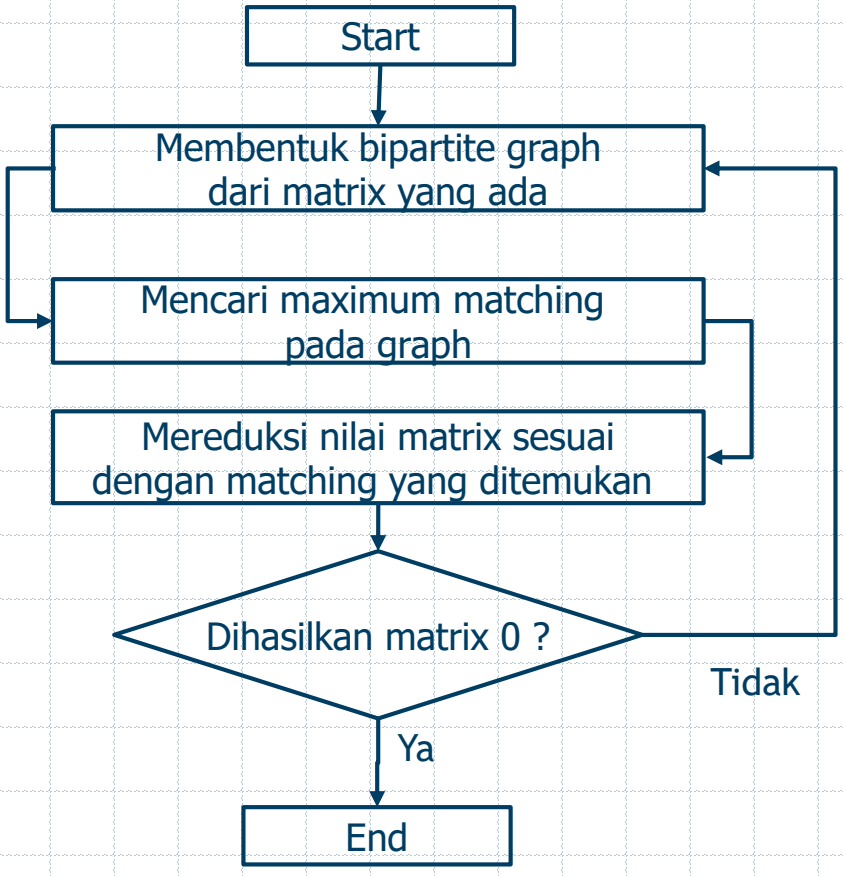
0	0	2	0	0
0	1	0	0	1
2	0	0	0	0
0	1	0	1	0
0	0	0	1	1



$m = \{(1,3), (2,2), (3,1), (4,4), (5,5)\}.$

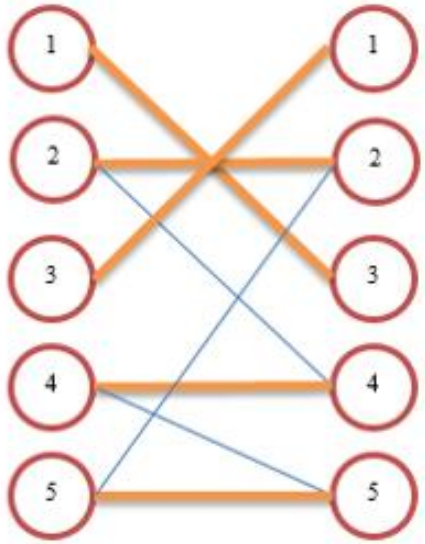
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 7

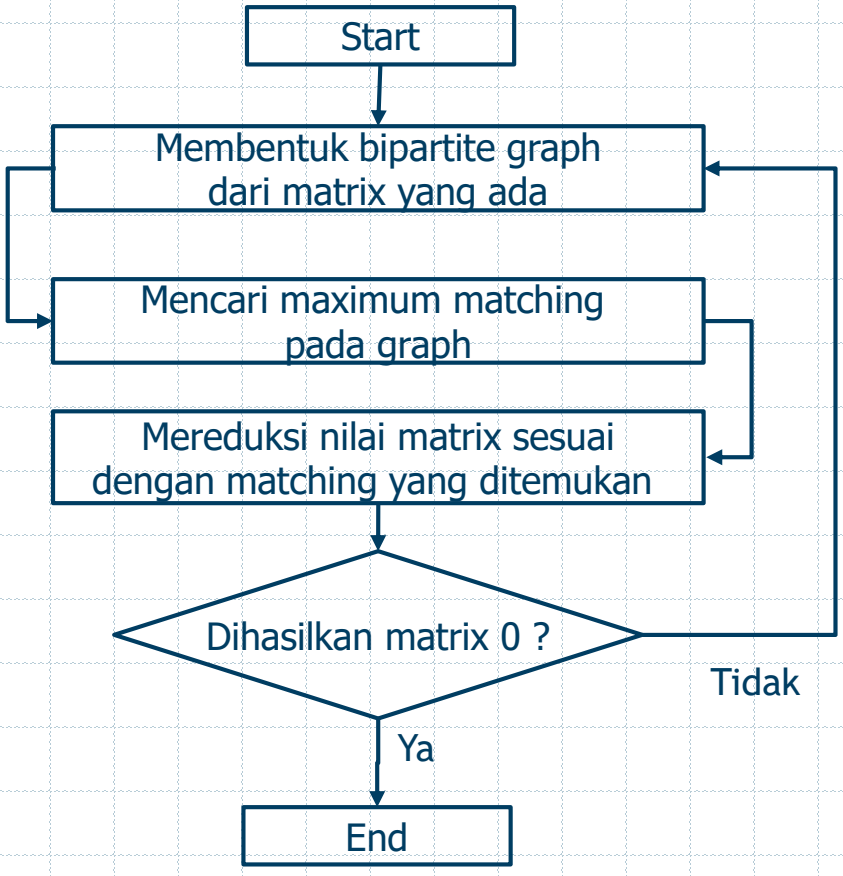
0	0	1	0	0
0	0	0	0	1
1	0	0	0	0
0	1	0	0	0
0	0	0	1	0



$m = \{(1,3), (2,2), (3,1), (4,4), (5,5)\}.$

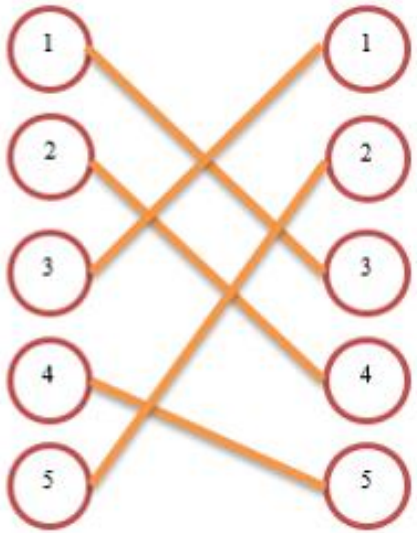
STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

Mencari *maximum bipartite matching*



Iterasi - 8

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0



$$m = \{(1,3), (2,4), (3,1), (4,5), (5,2).$$

STRATEGI PENYELESAIAN YET ANOTHER ASSIGNMENT PROBLEM

- Susunan lengkap dari penugasan tiap satuan waktunya berdasarkan proses diatas ditunjukkan oleh Tabel berikut.

Menit	Pekerja 1	Pekerja 2	Pekerja 3
1	1	0	0
2	1	0	0
3	0	1	0
4	0	1	0
5	2	0	1
6	0	2	1
7	0	2	1
8	0	0	1

- Berdasarkan tabel tersebut, waktu yang dibutuhkan untuk menyelesaikan semua pekerjaan tersebut adalah 8 menit serta didapatkan hasil penugasan pada menit pertama adalah 1, 0, 0.

1. PENDAHULUAN
2. TINJAUAN PUSTAKA
3. STRATEGI PENYELESAIAN
4. UJI COBA & EVALUASI
5. KESIMPULAN

UJI COBA KEBENARAN YET ANOTHER ASSIGNMENT PROBLEM

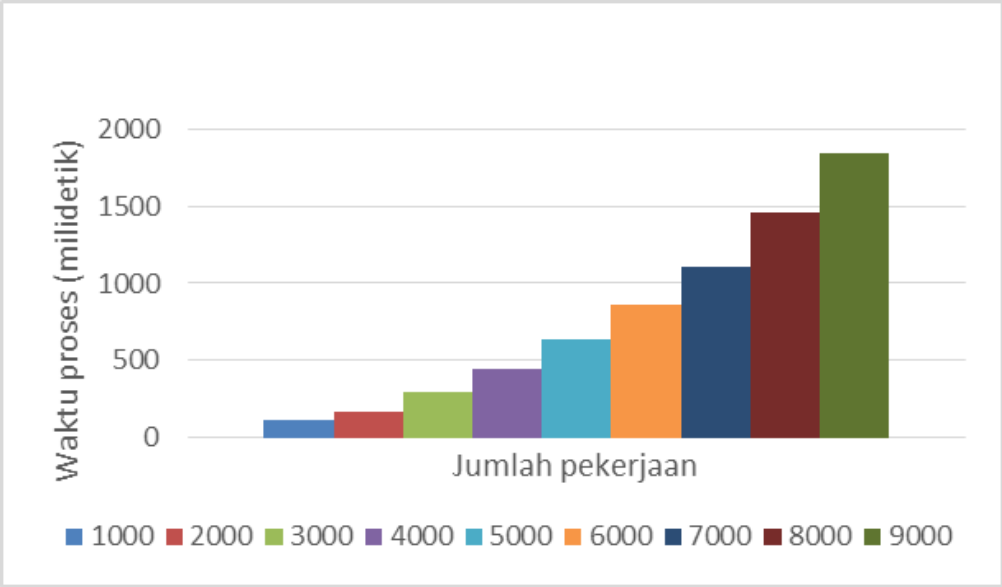
- Implementasi yang telah dilakukan pada permasalahan ***Yet Another Assignment Problem*** mendapat umpan balik ***accepted*** dari situs penilaian dari SPOJ.

17014913	2016-05-30 09:29:40	Yet Another Assignment Problem	**accepted**	0.08	64M	C++ 4.3.2

- Waktu minimum yang dibutuhkan program sebesar 0.08 detik, waktu maksimum sebesar 0.12 detik dan waktu rata-rata sebesar 0.103 detik. Memori yang dibutuhkan program konstan sebesar 64 MB.

UJI COBA EFISIENSI WAKTU YET ANOTHER ASSIGNMENT PROBLEM

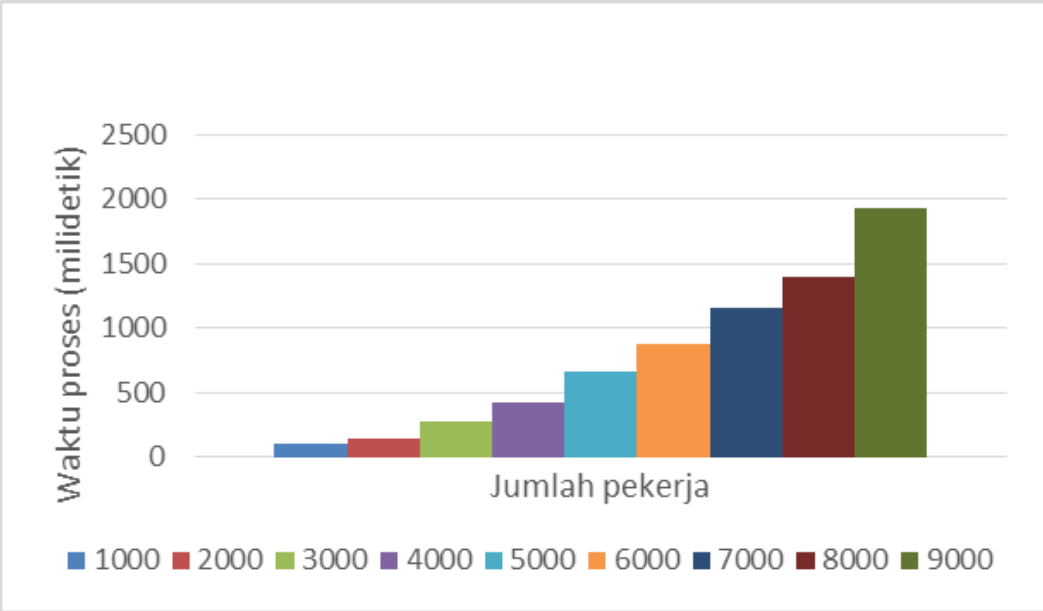
- Banyak pekerjaan dibuat bervariasi antara 1000 hingga 9000, jumlah pekerja ditetapkan konstan sebanyak 1000.



- Grafik cenderung mendekati kurva kuadratik, hal ini berarti waktu eksekusi program dipengaruhi secara kuadratik oleh jumlah pekerjaan.

UJI COBA EFISIENSI WAKTU YET ANOTHER ASSIGNMENT PROBLEM

- Banyak pekerja dibuat bervariasi antara 1000 hingga 9000, jumlah pekerjaan ditetapkan konstan sebanyak 1000.



- Grafik cenderung mendekati kurva kuadratik, hal ini berarti waktu eksekusi program dipengaruhi secara kuadratik oleh jumlah pekerja.

1. PENDAHULUAN
2. TINJAUAN PUSTAKA
3. STRATEGI PENYELESAIAN
4. UJI COBA & EVALUASI
5. KESIMPULAN

KESIMPULAN

- Implementasi algoritma Hopcroft-Karp dengan representasi *Bipartite Graph* dapat menyelesaikan permasalahan *Yet Another Assigment Problem* dengan benar.
- Kompleksitas waktu yang dibutuhkan untuk seluruh proses sistem adalah $O((m+n)^2)$. Pada m buah pekerjaan dan n orang pekerja.

ITS
Institut
Teknologi
Sepuluh Nopember

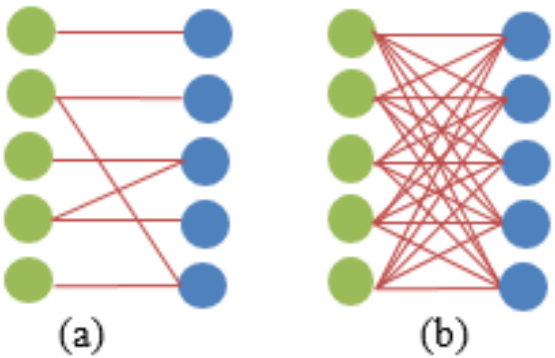
TERIMA KASIH

SARAN

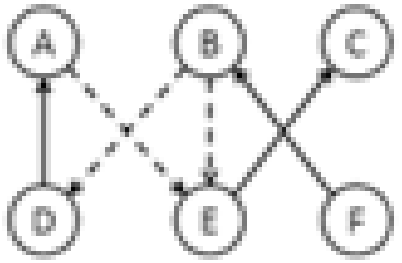
- Saran yang diberikan dalam pengembangan algoritma penyelesaian masalah penugasan adalah agar pada penelitian selanjutnya algoritma yang digunakan memiliki kompleksitas waktu yang lebih kecil dari $O((m+n)^2)$.

DEFINISI UMUM

- *Bipartite* graf merupakan graf spesial yang *vertex*-nya dapat dibagi menjadi dua himpunan *vertex* X dan Y dimana pada setiap himpunan *vertex*-nya, tidak ada *vertex* yang saling bertetangga.
- Sebuah matching pada graf $G = (V, E)$ adalah sebuah himpunan *edge* $M \subseteq E$ dimana tidak ada *edge* yang saling bersentuhan di dalam M



(a) Graf *Bipartite* Standar
(b) Graf *Bipartite Complete*

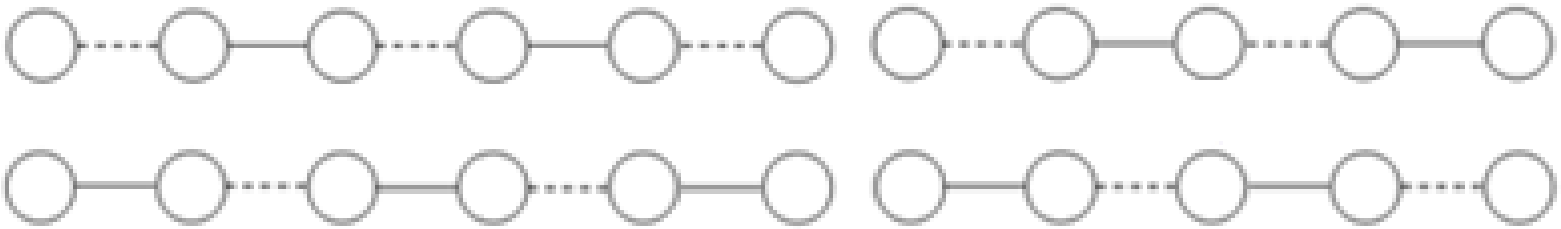


Matching Pada Sebuah Graf

DEFINISI UMUM

- *Alternating path* dari sebuah *matching* M adalah *path* dimana urutan *edge*-nya terdiri dari *matched edge* dan *unmatched edge* secara bergantian.

- *Augmenting path* dari sebuah *matching* M adalah *alternating path* yang dimulai dari sebuah *unmatched vertex* dan berakhir pada *unmatched vertex* pula.



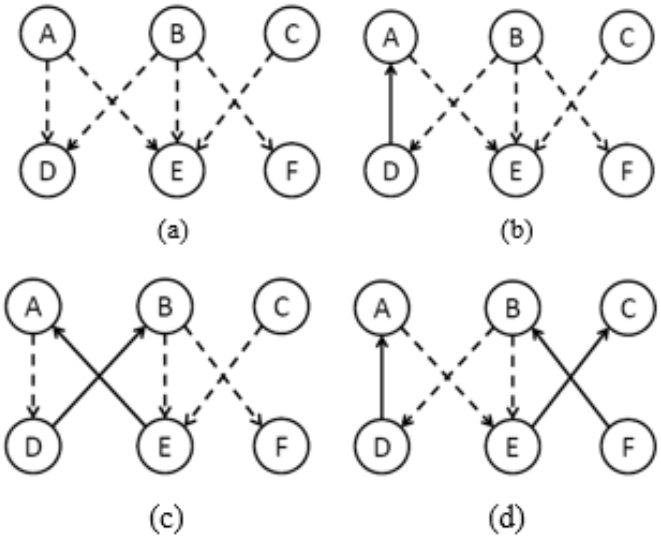
Alternating path dan *Augmenting path*

BIPARTITE MATCHING

- *Bipartite matching* merupakan kasus *matching* khusus yang melibatkan struktur *bipartite* graf didalamnya. *Bipartite* graf $G = (X \cup Y, E)$ dapat memiliki sebuah *complete matching* dengan kardinalitas $\min\{|X|, |Y|\}$
- *Pencarian maximum-size matching* dapat dibentuk secara iteratif melalui proses sebagi berikut:
  1. Dimulai dengan matching kosong, $M = \emptyset$.
  2. Mencari augmenting path dari matching yang sekarang sudah di dapatkan.
  3. Pada setiap iterasinya didapatkan sebuah matching baru yang memiliki kardinalitas $|M_{(i+1)}| = |M_i| + 1$ dengan cara menjadikan unmatched edge pada augmenting path yang telah didapatkan sebagai matched edge dan juga sebaliknya.

BIPARTITE MATCHING

4. Iterasi dilakukan hingga tidak ada lagi augmenting path yang dapat dibentuk dari matching yang terbaru.



Ilustrasi pencarian *maximum-size matching* sebuah graf