



UNIVERSITAS INDONESIA

**ALGORITMA PELABELAN TOTAL SIMPUL AJAIB
PADA GRAF LINGKARAN, MATAHARI, DAN KECEBONG**

SKRIPSI

**ALFA ISTI ANANDA
0606067244**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
PROGRAM STUDI SARJANA MATEMATIKA
DEPOK
JULI 2010**



UNIVERSITAS INDONESIA

**ALGORITMA PELABELAN TOTAL SIMPUL AJAIB
PADA GRAF LINGKARAN, MATAHARI, DAN KECEBONG**

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana sains

**ALFA ISTI ANANDA
0606067244**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
PROGRAM STUDI SARJANA MATEMATIKA
DEPOK
JULI 2010**

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.



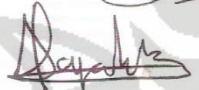

Nama : Alfa Isti Ananda
NPM : 0606067244
Tanda Tangan : 
Tanggal : 8 Juli 2010

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :
Nama : Alfa Isti Ananda
NPM : 0606067244
Program Studi : Sarjana Matematika
Judul Skripsi : Algoritma Pelabelan Total Simpul Ajaib pada Graf
Lingkaran, Matahari, dan Kecebong

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Sains pada Program Studi Sarjana Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Dra. Denny R. Silaban, M.Kom. ()
Pembimbing : Dhian Widya, S.Si, M.Kom. ()
Penguji : Dr. Al Haji Akbar B., M.Sc. ()
Penguji : Sarini Abdullah, M.Stats. ()

Ditetapkan di : Depok
Tanggal : 8 Juli 2010

KATA PENGANTAR

Puji syukur kepada Allah SWT atas segala berkah dan karunia-Nya sehingga penulisan skripsi ini selesai tepat pada waktunya.

Penulis menyadari bahwa penulisan skripsi ini selesai atas bantuan dan dukungan dari berbagai pihak. Oleh karena itu, penulis menghaturkan terima kasih kepada:

- (1) Dra. Denny Riama Silaban, M.Kom. dan Dhian Widya, S.Si, M.Kom. selaku pembimbing skripsi yang telah banyak meluangkan waktu untuk membimbing, memberi saran, dan memberi bantuan untuk penulis dalam menyelesaikan skripsi ini.
- (2) Dr. Kiki Ariyanti Sugeng, Dra. Siti Aminah, M.Kom., Bevina D. Handari, PhD., Prof. Dr. Djati Kerami, dan Dra. Nora Hariadi, M.Si. yang telah memberikan saran, nasihat, serta semangat kepada penulis selama pembuatan skripsi ini.
- (3) Dra. Suarsih Utama selaku pembimbing akademis penulis selama menjalani masa kuliah dan terima kasih untuk seluruh dosen beserta staf Departemen Matematika FMIPA UI atas kesabaran dan bimbingannya.
- (4) Mama, Papa, Bobi, Embah Putri, Om, Tante, dan seluruh saudara penulis yang selalu memberikan doa, semangat, dan dukungan bagi penulis.
- (5) Mella, Milla, Tami yang telah berjuang bersama selama penyusunan skripsi ini. Terima kasih atas bantuan yang kalian berikan, terutama untuk Mella dan Tami yang juga membantu dalam *running* program matahari yang tidak pernah diskon waktunya.
- (6) Widya, Latief, dan Rita Yuliana yang bernasib sama. Sama-sama terlempar dari peminatan semasa kuliah.
- (7) Oppie dan Ar Rizqi yang telah membantu penulis dalam penyusunan skripsi. Sukses untuk kalian!!

- (8) Bekti, Kak Akmal, dan Anto (FT 2006) yang menjadi tempat bertanya penulis ketika laptop rusak disaat-saat genting, salah satunya sebelum maju SIG 2. Bukan hanya tempat bertanya, tapi juga sebagai tukang servis laptop.
- (9) Teguh yang bersedia dijarah laptopnya untuk *running* program (walaupun hasilnya hilang bersama *flashdisk* pink). Terima kasih juga atas doa dan semangat yang diberikan.
- (10) Noor dan Puspa yang setia bersama penulis selama perkuliahan. Terima kasih atas semangat, masukan, serta omelan yang pernah diberikan. *Maybe friendship not make people always together, not make we always contact each other.. But I know that friendship keep our heart always together..*
- (11) Lena, Farah, Yunita, Dian, Nanad, Nisa, Qq, Nita, Nurgi, Helmet, Ita, dan anak-anak geng bekel++ lainnya. Terima kasih atas persahabatan yang telah terjalin selama ini. *In my life, I learned how to love, how to smile, how to be strong, how to survive, how to work hard, but i didn't learn how to stop loving sister like you..*
- (12) Lee, Yuri, Reza, Rifza, Yuko, Hot, Tika, Oza, Rama, Tisna, Tino, Rita P. dan teman-teman yang telah lulus semester lalu (Inne, Mei, Anggha), serta teman-teman angkatan 2006 lainnya. Terima kasih atas kebersamaannya.
- (13) Teman-teman Matematika UI, khususnya angkatan 2004 (*esp.* mentorku Kak Intan), 2005 (*esp.* Kak Anggie dan Kak Wicha), 2007 (*esp.* Stefi, Dita, dan Arif), dan 2008.
- (14) Teman-teman LBC beserta pengajarnya yang cantik, Miss Audrey. Terima kasih atas semangatnya.

Penulis juga mengucapkan terima kasih kepada pihak yang tidak dapat disebutkan satu per satu, yang telah membantu dalam penyusunan skripsi ini. Akhir kata, penulis mohon maaf apabila terdapat kesalahan atau kekurangan dalam skripsi ini. Semoga skripsi ini membawa manfaat bagi perkembangan ilmu.

Penulis
2010

**HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS**

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

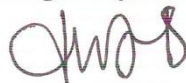
Nama : Alfa Isti Ananda
NPM : 0606067244
Program Studi : Sarjana Matematika
Departemen : Matematika
Fakultas : Matematika dan Ilmu Pengetahuan Alam
Jenis karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty Free Right*) atas karya ilmiah saya yang berjudul :
Algoritma Pelabelan Total Simpul Ajaib pada Graf Lingkaran, Matahari, dan Kecebong

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/format-kan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : 8 Juli 2010
Yang menyatakan



(Alfa Isti Ananda)

ABSTRAK

Nama : Alfa Isti Ananda
Program Studi : Matematika
Judul : Algoritma Pelabelan Total Simpul Ajaib pada Graf Lingkaran, Matahari, dan Kecebong

Misalkan G adalah graf dengan himpunan simpul $V = V(G)$ dan himpunan busur $E = E(G)$, dimana $|V(G)|$ dan $|E(G)|$ menyatakan banyaknya simpul dan busur pada G . Suatu pemetaan λ dari $V \cup E$ ke himpunan bilangan bulat $1, 2, \dots, |V|+|E|$ disebut **pelabelan total simpul ajaib** pada G jika λ merupakan pemetaan bijektif dengan sifat bahwa untuk setiap simpul $v \in V$, $\lambda(v) + \sum_{u \in N(v)} \lambda(uv) = k$ dimana $N(v)$ adalah himpunan semua simpul yang bertetangga dengan v . Nilai k disebut konstanta ajaib dari λ . Algoritma pelabelan sembarang graf secara umum bersifat *NP-complete*. Baker dan Sawada telah memberikan algoritma pelabelan total simpul ajaib pada graf lingkaran C_n dan graf roda W_n . Pada skripsi ini, algoritma lingkaran tersebut akan dibahas. Selain itu, akan dibangun algoritma pelabelan total simpul ajaib pada graf matahari $C_n \odot \bar{K}_1$ dan graf kecebong $T_{m,n}$. Menggunakan algoritma-algoritma tersebut dapat dihasilkan semua pelabelan total simpul ajaib pada graf yang terkait. Algoritma-algoritma ini akan diimplementasikan menggunakan program. Sebagai hasil implementasi dilakukan simulasi yang memberikan banyaknya pelabelan total simpul ajaib yang berbeda dari graf lingkaran C_n dengan $3 \leq n \leq 10$, graf matahari $C_n \odot \bar{K}_1$ dengan $3 \leq n \leq 7$, dan graf kecebong $T_{m,n}$ dengan $3 \leq m \leq 7, 1 \leq n \leq 5$ untuk setiap nilai k yang mungkin.

Kata kunci : Pelabelan total simpul ajaib, graf lingkaran, graf matahari, graf kecebong
xiv+88 halaman ; 50 gambar; 9 tabel
Daftar Pustaka : 8 (1995-2010)

ABSTRACT

Name : Alfa Isti Ananda
Study Program: Mathematics
Title : Vertex Magic Total Labeling Algorithms on Cycles, Suns, and Tadpoles Graphs

Let graph G has vertex set $V = V(G)$ and edge set $E = E(G)$, and let $|V(G)|$ and $|E(G)|$ is the number of vertices and edges on G . A one-to-one map λ from $V \cup E$ onto $\{1, 2, \dots, |V|+|E|\}$ is a **vertex magic total labeling** if there is a constant k so that for every vertex $v \in V$, $\lambda(v) + \sum_{u \in N(v)} \lambda(uv) = k$ where $N(v)$ denoted the set of vertices adjacent to v . The constant k is called the magic constant of λ . In general, the labeling algorithms on any graphs is NP-complete. In their paper, Baker and Sawada give the vertex magic total labeling algorithms on cycle graph C_n and wheel graph W_n . This *skripsi* explains the vertex magic total labeling algorithm on cycle from Baker and Sawada and vertex magic total labeling algorithms on sun graph $C_n \odot \bar{K}_1$ and tadpole graph $T_{m,n}$. Using these algorithms, all non-isomorphic vertex magic total labelings on those classes of graphs can obtained. These algorithms are implemented as computer programs. From simulations, we get the number of non-isomorphic vertex magic total labelings on cycles C_n ($3 \leq n \leq 10$), suns $C_n \odot \bar{K}_1$ ($3 \leq n \leq 7$), and tadpoles $T_{m,n}$ ($3 \leq m \leq 7, 1 \leq n \leq 5$) for every possible value of k .

Key words : Vertex magic total labeling, cycle graph, sun graph, tadpole graph
xiv+88 pages ; 50 pictures; 9 tables
Bibliography : 8 (1995-2010)

DAFTAR ISI

HALAMAN PERNYATAAN ORISINALITAS.....	iii
HALAMAN PENGESAHAN.....	iv
KATA PENGANTAR	v
ABSTRAK	viii
ABSTRACT.....	ix
DAFTAR ISI.....	x
DAFTAR TABEL.....	xi
DAFTAR GAMBAR	xii
DAFTAR LAMPIRAN.....	xiv
1. PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Permasalahan.....	3
1.3 Tujuan.....	3
1.4 Pembatasan Masalah	3
1.5 Sistematika Penulisan.....	4
2. LANDASAN TEORI.....	5
2.1 Definisi dan Istilah dalam Teori Graf.....	5
2.2 Jenis-jenis Graf.....	8
2.3 Pelabelan Graf	9
3. ALGORITMA PTSA UNTUK GRAF LINGKARAN, MATAHARI, DAN KECEBONG	16
3.1 Algoritma PTSA untuk Graf Lingkaran.....	19
3.2 Algoritma PTSA untuk Graf Matahari.....	30
3.3 Algoritma PTSA untuk Graf Kecebong.....	43
4. IMPLEMENTASI DAN SIMULASI ALGORITMA PTSA.....	56
4.1 Implementasi dan Simulasi Algoritma PTSA untuk Graf Lingkaran	57
4.2 Implementasi dan Simulasi Algoritma PTSA untuk Graf Matahari	63
4.3 Implementasi dan Simulasi Algoritma PTSA untuk Graf Kecebong.....	67
5. KESIMPULAN.....	76
DAFTAR PUSTAKA	78

DAFTAR TABEL

Tabel 4. 1	Banyaknya PTSA tidak isomorfik untuk graf lingkaran C_n dengan $n = 3$ s.d. 10 berdasarkan $\frac{5n+3}{2} \leq k \leq \frac{7n+3}{2}$	61
Tabel 4. 2	Banyaknya PTSA tidak isomorfik untuk graf matahari $C_n \odot \bar{K}_1$ dengan $n = 3$ s.d. 7 berdasarkan $5n + 2 \leq k \leq 6n + 1$	65
Tabel 4. 3	Banyaknya PTSA tidak isomorfik untuk graf $T_{3,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-10}{2(m+n-1)}\right)$..	70
Tabel 4. 4	Banyaknya PTSA tidak isomorfik untuk graf $T_{4,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-10}{2(m+n-1)}\right)$..	70
Tabel 4. 5	Banyaknya PTSA tidak isomorfik untuk graf $T_{5,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-10}{2(m+n-1)}\right)$..	71
Tabel 4. 6	Banyaknya PTSA tidak isomorfik untuk graf $T_{6,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-10}{2(m+n-1)}\right)$..	71
Tabel 4. 7	Banyaknya PTSA tidak isomorfik untuk graf $T_{7,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-10}{2(m+n-1)}\right)$..	72
Tabel 4. 8	Eksistensi dari PTSA graf $T_{m,n}$ berdasarkan simulasi program dengan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-10}{2(m+n-1)}\right)$..	72
Tabel 5. 1	Eksistensi PTSA untuk graf Lingkaran C_n , Matahari $C_n \odot \bar{K}_1$, dan Kecebong $T_{m,n}$	76

DAFTAR GAMBAR

Gambar 2. 1	Contoh Graf G	6
Gambar 2. 2	Dua graf G dan G' yang isomorfik.....	7
Gambar 2. 3	(a) P_5 , (b) C_5	8
Gambar 2. 4	(a) $C_5 \odot \bar{K}_1$, (b) $T_{5,3}$	9
Gambar 2. 5	(a) Pelabelan simpul, (b) Pelabelan busur, (c) Pelabelan Total ...	10
Gambar 2. 6	PTSA C_3 dengan $k = 9$ (a) dan dualnya dengan $k = 12$ (b)	12
Gambar 2. 7	Dua PTSA yang isomorfik	13
Gambar 3. 1	Graf C_4	19
Gambar 3. 2	Proses pembentukan PTSA pada graf C_n	21
Gambar 3. 3	PTSA C_4 yang isomorfik dengan kasus rotasional simetris.....	24
Gambar 3. 4	PTSA C_4 yang isomorfik dengan kasus refleksional simetris.....	25
Gambar 3. 5	Proses pada <i>initializeCycle</i> untuk membentuk PTSA C_3 dengan $k = 10$	26
Gambar 3. 6	<i>Backtracking</i> pada <i>initializeCycle</i>	27
Gambar 3. 7	Proses pada <i>extendCycle(2)</i> untuk membentuk PTSA C_3 dengan $k = 10$	28
Gambar 3. 8	<i>Backtracking</i> pada <i>extendCycle(2)</i> dan <i>extendCycle(3)</i> diproses untuk membentuk PTSA C_3 dengan $k = 10$	29
Gambar 3. 9	Graf $C_3 \odot \bar{K}_1$	31
Gambar 3. 10	Proses pembentukan PTSA pada graf $C_n \odot \bar{K}_1$	34
Gambar 3. 11	PTSA $C_3 \odot \bar{K}_1$ yang isomorfik dengan kasus rotasional	35
Gambar 3. 12	PTSA $C_3 \odot \bar{K}_1$ yang isomorfik dengan kasus refleksional	36
Gambar 3. 13	Proses pada <i>initializeSun</i> untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$	38
Gambar 3. 14	Proses pada <i>extendSun(2)</i> untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$	39
Gambar 3. 15	Proses pada <i>extendSun(3)</i> untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$	40
Gambar 3. 16	<i>Backtracking</i> pada <i>extendSun(3)</i>	41
Gambar 3. 17	Proses pada <i>extendSun(4)</i> untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$	42
Gambar 3. 18	Graf $T_{5,2}$	43
Gambar 3. 19	Proses pembentukan PTSA pada graf $T_{m,n}$	47
Gambar 3. 20	PTSA isomorfik untuk graf $T_{5,2}$	48
Gambar 3. 21	Proses pada <i>initializeTadpole</i> untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$	49
Gambar 3. 22	Proses pada <i>extendTadpole(1)</i> untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$	50
Gambar 3. 23	<i>Backtracking</i> pada <i>extendTadpole(1)</i>	51

Gambar 3. 24	Proses pada <i>extendTadpole(2)</i> dan <i>extendTadpole(3)</i> untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$	52
Gambar 3. 25	Proses pada <i>finalizeTadpole(2)</i> untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$	53
Gambar 3. 26	<i>Backtracking</i> pada <i>finalizeTadpole(3)</i>	53
Gambar 3. 27	Proses pada <i>finalizeTadpole(2)</i> dan <i>finalizeTadpole(3)</i> untuk memmentuk PTSA $T_{3,3}$ dengan $k = 17$	54
Gambar 4. 1	Tampilan <i>Command Window</i> awal	57
Gambar 4. 2	Tampilan masukan (a) dan keluaran (b) untuk PTSA C_3 dengan $k = 10$	58
Gambar 4. 3	PTSA C_3 dengan $k = 10$	59
Gambar 4. 4	PTSA-PTSA tidak isomorfik pada graf C_6	60
Gambar 4. 5	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf C_n seiring dengan bertambahnya nilai n	62
Gambar 4. 6	Tampilan masukan (a) dan keluaran (b) untuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$	64
Gambar 4. 7	PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$	65
Gambar 4. 8	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $C_4 \odot \bar{K}_1$ seiring dengan bertambahnya nilai n	66
Gambar 4. 9	Tampilan masukan (a) dan keluaran (b) untuk PTSA $T_{3,3}$ dengan $k = 17$	68
Gambar 4. 10	5 PTSA tidak isomorfik pada graf $T_{3,3}$ dengan $k = 17$	69
Gambar 4. 11	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{3,n}$ dan $T_{4,n}$ seiring dengan bertambahnya nilai n	73
Gambar 4. 12	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{5,n}$ dan $T_{6,n}$ seiring dengan bertambahnya nilai n	73
Gambar 4. 13	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{7,n}$ seiring dengan bertambahnya nilai n	74
Gambar 4. 14	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{m,1}$ dan $T_{m,2}$ seiring dengan bertambahnya nilai m	74
Gambar 4. 15	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{m,3}$ dan $T_{m,4}$ seiring dengan bertambahnya nilai m	75
Gambar 4. 16	Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{m,5}$ seiring dengan bertambahnya nilai m	75

DAFTAR LAMPIRAN

- Lampiran 1 Tampilan masukan dan keluaran untuk PTSA C_679
Lampiran 2 *Listing* program PTSA graf lingkaran (CD)
Lampiran 3 *Listing* program PTSA graf matahari (CD)
Lampiran 4 *Listing* program PTSA graf kecebong (CD)



BAB 1

PENDAHULUAN

1.1 Latar Belakang

Teori graf adalah bagian dari matematika diskrit yang banyak digunakan sebagai alat bantu untuk menggambarkan atau menyatakan suatu persoalan agar lebih mudah dimengerti dan diselesaikan. Banyak persoalan akan menjadi lebih jelas untuk diterangkan jika dapat direpresentasikan dalam bentuk graf.

Suatu graf G didefinisikan sebagai pasangan himpunan $(V(G), E(G))$, dimana $V(G)$ adalah himpunan tak kosong dari simpul-simpul dan $E(G)$ adalah himpunan busur-busur yang menghubungkan simpul-simpul pada $V(G)$. Busur merupakan pasangan tak-terurut dari simpul-simpul pada $V(G)$. Banyaknya anggota pada himpunan simpul $V(G)$ atau V dinyatakan sebagai $|V|$. Sedangkan banyaknya anggota pada himpunan busur $E(G)$ atau E dinyatakan sebagai $|E|$. Graf G dikatakan berhingga jika $|V|$ berhingga.

Terdapat graf dengan ciri-ciri tertentu yang diberi nama khusus seperti graf lintasan, graf lingkaran, graf matahari, dan graf kecebong. Graf lintasan (*Path graph*), P_n , adalah graf dengan n simpul yang mempunyai busur-busur $v_1v_2, v_2v_3, \dots, v_{n-1}v_n$. Graf lingkaran (*Cycle graph*), C_n , adalah graf yang diperoleh dari graf lintasan dengan penambahan busur antara $v_n v_1$. Graf matahari (*Sun graph*), $C_n \odot \bar{K}_1$, adalah graf yang diperoleh dari graf lingkaran C_n dengan penambahan satu simpul berderajat satu di setiap simpul graf C_n . Graf kecebong (*Tadpole graph*), $T_{m,n}$, adalah graf yang diperoleh dari graf lingkaran C_m dan graf lintasan P_n dengan penambahan satu busur yang menghubungkan salah satu simpul pada graf lingkaran dan simpul awal (atau simpul akhir) pada graf lintasan.

Pelabelan pada graf G adalah pemetaan bijektif dari himpunan simpul V atau himpunan busur E , atau keduanya $(V \cup E)$ ke suatu himpunan, biasanya merupakan himpunan bilangan bulat positif, dengan kondisi tertentu. Pelabelan yang dibahas pada skripsi ini adalah pemetaan bijektif dari $V \cup E$ ke himpunan bilangan bulat positif berurutan yang dimulai dari 1. Jika daerah asal dari pemetaan hanya himpunan simpul, maka pelabelan disebut pelabelan simpul. Jika daerah

asal hanya himpunan busur, maka pelabelan disebut pelabelan busur. Jika daerah asal merupakan gabungan dari himpunan simpul dan busur, maka pelabelan disebut pelabelan total. Pada pelabelan total, jumlah dari semua label yang terkait dengan suatu elemen tertentu dari graf disebut bobot. Bobot dapat dihitung untuk simpul dan busur. Bobot busur adalah penjumlahan dari label busur dan label simpul-simpul yang dihubungkan dengan busur tersebut. Bobot simpul adalah penjumlahan dari label simpul dan label busur-busur yang hadir pada simpul tersebut. Graf G dikatakan memiliki pelabelan ajaib jika bobot untuk setiap simpul dan/atau busur bernilai sama.

Dalam skripsi ini hanya akan dibahas mengenai pelabelan total simpul ajaib. Pelabelan total simpul ajaib (PTSA) adalah pelabelan pada graf sedemikian sehingga bobot setiap simpulnya sama, yaitu bernilai k . Bilangan k inilah yang disebut sebagai konstanta ajaib.

Dua graf G dan G' yang diberi label dikatakan memiliki pelabelan isomorfik apabila terdapat pemetaan bijektif dari label-label untuk simpul-simpul pada G ke label-label untuk simpul-simpul pada G' dengan sifat bahwa dua simpul berlabel pada G bertetangga jika dan hanya jika peta dari kedua simpul berlabel tersebut juga bertetangga di G' .

Telah terdapat hasil-hasil mengenai eksistensi dari PTSA untuk kelas-kelas graf tertentu yang telah dipublikasikan. Misalnya, untuk graf lingkaran, MacDougall *et al.* (2002) telah membuktikan bahwa graf tersebut memiliki PTSA. Untuk graf matahari, Slamin *et al.* (2006) telah membuktikan bahwa gabungan tak-terhubung t graf matahari yang isomorfik memiliki PTSA untuk $t \geq 1$. Untuk gabungan tak-terhubung t graf matahari yang tidak harus isomorfik, Rahim dan Slamin (2008) telah membuktikan bahwa gabungan graf matahari tersebut memiliki PTSA untuk $t \geq 1$. Namun, kedua hasil yang diperoleh untuk gabungan graf matahari tersebut hanya untuk nilai k tertentu, bukan untuk semua nilai k yang mungkin.

Pendekatan lain yang dapat dilakukan untuk memperoleh semua PTSA yang mungkin adalah menggunakan algoritma. Dengan algoritma dapat dicari semua PTSA tidak isomorfik untuk setiap nilai k yang mungkin. Dalam makalah Baker dan Sawada (2008), telah dibahas algoritma untuk membangun PTSA yang

tidak isomorfik pada graf lingkaran. Pembahasan pada skripsi ini adalah mengulas ulang algoritma tersebut, kemudian dikembangkan untuk membangun algoritma-algoritma PTSA pada kelas-kelas graf yang mengandung graf lingkaran dan lintasan, seperti graf matahari dan kecebong. Sampai saat ini, belum ada hasil penelitian yang menyatakan bahwa graf kecebong memiliki suatu PTSA.

1.2 Permasalahan

Bagimanakah cara membangun algoritma untuk memperoleh semua PTSA yang berbeda pada kelas-kelas graf yang mengandung graf lingkaran dan/atau lintasan?

1.3 Tujuan

Memberikan algoritma-algoritma PTSA untuk kelas-kelas graf yang mengandung graf lingkaran dan/atau lintasan. Algoritma-algoritma ini akan diimplementasikan menggunakan program MATLAB. Sebagai hasil implementasi, dilakukan simulasi yang memberikan banyaknya PTSA berbeda untuk semua nilai k yang mungkin dari graf terkait.

1.4 Pembatasan Masalah

Dalam skripsi ini hanya dibatasi pada pembangunan algoritma-algoritma PTSA pada graf lingkaran, matahari, dan kecebong. Algoritma-algoritma ini akan diimplementasikan menggunakan program MATLAB sehingga dapat diperoleh banyaknya PTSA yang tidak isomorfik pada graf yang terkait untuk semua nilai k yang mungkin.

1.5 Sistematika Penulisan

Skripsi ini terbagi menjadi 5 bab. Pada Bab 2 diberikan definisi-definisi dan konsep dasar mengenai teori graf dan pelabelan graf. Pada Bab 3 diberikan algoritma-algoritma PTSA untuk graf lingkaran, matahari, dan kecebong. Bab 4 memberikan implementasi dan simulasi dari algoritma-algoritma yang telah diberikan. Bab 5 merupakan Bab kesimpulan yang memberikan jawaban dari permasalahan dan pencapaian tujuan penelitian.



BAB 2 LANDASAN TEORI

Pada Bab ini akan diberikan beberapa definisi dan konsep dasar dalam teori graf dan pelabelan graf yang akan digunakan pada bab selanjutnya.

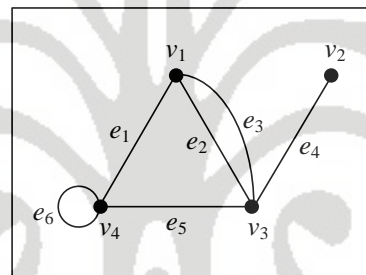
2.1 Definisi dan Istilah dalam Teori Graf

Suatu **graf** $G = (V, E)$ terdiri dari gabungan himpunan tak kosong dari **simpul** (*vertices*) $V(G)$ dan himpunan **busur** (*edges*) $E(G)$ yang menghubungkan simpul-simpul pada $V(G)$. Himpunan busur dapat berupa himpunan kosong (disebut juga **graf kosong**). Jika suatu busur menghubungkan satu simpul ke simpul itu sendiri, maka busur tersebut disebut **gelung** (*loop*). Jika dua simpul dihubungkan oleh lebih dari satu busur, maka busur tersebut disebut **busur berganda** (*multiple edges*). Graf yang tidak mengandung gelang dan busur berganda disebut **graf sederhana** (*simple graph*). Untuk penyederhanaan, $G = (V, E)$ akan dinotasikan dengan G , $V(G)$ dengan V , dan $E(G)$ dengan E . Banyaknya simpul pada graf G dinyatakan sebagai $|V|$ dan banyaknya busur dinyatakan sebagai $|E|$. Apabila $|V|$ berhingga, maka graf G disebut **graf berhingga** (*finite graph*).

Suatu graf G disebut **graf berarah** (*directed graph*) bila anggota himpunan busur dari graf G berupa **busur berarah** (*arc*), yaitu pasangan terurut dari simpul-simpul di V . Apabila anggota himpunan busur pada graf G bukan merupakan pasangan terurut dari simpul-simpul, maka graf G disebut **graf tak berarah** (*undirected graph*). Untuk selanjutnya, diberikan definisi-definisi yang terkait dengan graf tak berarah.

Dua simpul pada suatu graf G dikatakan **bertetangga** (*adjacent*) apabila terdapat satu atau lebih busur yang menghubungkan kedua simpul tersebut. Suatu busur dikatakan **hadir** (*incident*) pada suatu simpul bila simpul itu merupakan salah satu ujung dari busur tersebut, sehingga simpul-simpul yang dihubungkan oleh suatu busur disebut **simpul ujung** (*endpoint*) dari busur. Dua busur pada graf G dikatakan bertetangga jika kedua busur tersebut hadir pada simpul yang

sama. Jumlah busur yang hadir pada suatu simpul v disebut **derajat** (*degree*) dari simpul v (ditulis $d(v)$). Jika suatu simpul v tidak bertetangga dengan simpul yang lain, maka simpul tersebut memiliki $d(v) = 0$. Simpul dengan derajat 0 disebut sebagai **simpul terpencil** (*isolated vertex*). Jika suatu simpul v bertetangga dengan tepat satu simpul lain, maka simpul tersebut memiliki $d(v) = 1$. Simpul dengan derajat 1 disebut sebagai **simpul terminal** (*terminal vertex*). Jika terdapat gelang pada suatu simpul, maka gelang dapat dikatakan sebagai busur yang hadir dua kali untuk menghitung derajat dari simpul tersebut. Derajat terkecil dari suatu graf G dinyatakan dengan $\delta = \delta(G) = \min_{v \in V} d(v)$ dan derajat terbesar dinyatakan dengan $\Delta = \Delta(G) = \max_{v \in V} d(v)$. Jika $\delta = \Delta = r$, maka graf G disebut sebagai **graf teratur berderajat r** (*r -regular graph*).

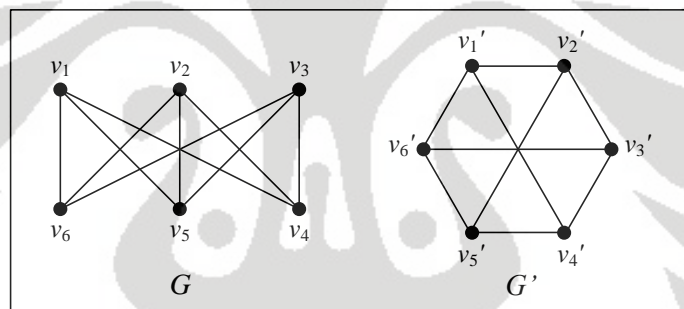


Gambar 2.1 Contoh Graf G

Suatu graf dapat direpresentasikan dalam bentuk gambar, dimana simpul-simpul direpresentasikan sebagai titik-titik dan busur-busur direpresentasikan sebagai segmen garis yang menghubungkan simpul-simpul. Biasanya, simpul ditulis sebagai v , sedangkan busur ditulis sebagai e atau sebagai pasangan kedua simpul ujung, $v_i v_j$, $i, j = 1, 2, \dots, |V|$. Pada Gambar 2.1 diberikan contoh graf G dengan himpunan simpul $V = \{v_1, v_2, v_3, v_4\}$ dan himpunan busur $E = \{e_1, e_2, e_3, e_4, e_5, e_6\} = \{v_1 v_4, v_1 v_3, v_1 v_3, v_2 v_3, v_3 v_4, v_4 v_4\}$. Nilai $|V|$ dan $|E|$ pada graf G masing-masing adalah 4 dan 6. Graf G bukan graf sederhana karena mengandung busur berganda yang ditunjukkan oleh busur e_2, e_3 dan gelang yang ditunjukkan oleh e_6 . Simpul v_1 dan v_4 bertetangga karena dihubungkan oleh busur e_1 . Busur e_1 dan e_2 bertetangga karena hadir pada simpul yang sama, yaitu simpul v_1 . Derajat setiap simpul pada Gambar 2.1 adalah $d(v_1) = 3$, $d(v_2) = 1$, $d(v_3) = 4$, $d(v_4) = 4$. Oleh

karena $d(v_2) = 1$, maka v_2 disebut simpul terminal. Graf G bukan graf teratur karena $\delta = 1$ dan $\Delta = 4$.

Dua graf $G = (V, E)$ dan $G' = (V', E')$ dikatakan **isomorfik** apabila terdapat pemetaan bijektif f dari V ke V' dengan sifat bahwa simpul v_1 dan v_2 bertetangga di G jika dan hanya jika simpul $f(v_1)$ dan $f(v_2)$ bertetangga di G' . Hal ini berlaku untuk setiap simpul di G . Akibat dari pemetaan bijektif ini adalah dua graf yang isomorfik harus memiliki jumlah simpul yang sama. Lebih jauh lagi, graf yang isomorfik juga harus memiliki jumlah busur yang sama karena pemetaan bijektif dari V ke V' dengan sifat yang telah disebutkan juga menjamin pemetaan bijektif dari E ke E' dengan sifat bahwa jika dua busur bertetangga di G , maka peta dari busur tersebut di G' juga bertetangga. Derajat dari simpul-simpul pada dua graf yang isomorfik juga harus sama, yaitu simpul v dengan derajat $d(v)$ di G berkorespondensi dengan simpul $f(v)$ dengan derajat $d(f(v)) = d(v)$ di G' .



Gambar 2.2 Dua graf G dan G' yang isomorfik

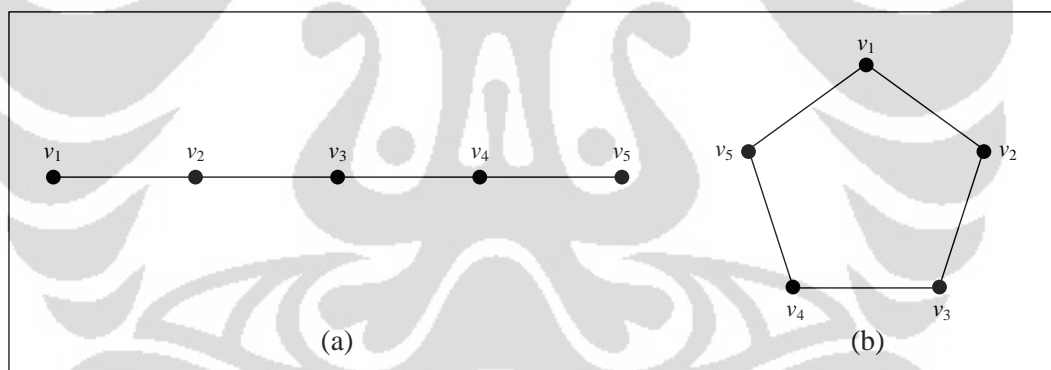
Dua graf yang diberikan pada Gambar 2.2 adalah graf yang isomorfik karena terdapat pemetaan bijektif f dari V ke V' dengan aturan $f(v_1) = v_1'$, $f(v_2) = v_3'$, $f(v_3) = v_5'$, $f(v_4) = v_6'$, $f(v_5) = v_4'$, $f(v_6) = v_2'$. Berdasarkan aturan ini terlihat bahwa jika simpul-simpul di G bertetangga, maka peta dari simpul-simpul tersebut di G' juga bertetangga. Aturan untuk simpul-simpul tersebut akan mengakibatkan $f(v_1v_4) = v_1'v_6'$, $f(v_1v_5) = v_1'v_4'$, $f(v_1v_6) = v_1'v_2'$, $f(v_2v_4) = v_3'v_6'$, $f(v_2v_5) = v_3'v_4'$, $f(v_2v_6) = v_3'v_2'$, $f(v_3v_4) = v_5'v_6'$, $f(v_3v_5) = v_5'v_4'$, dan $f(v_3v_6) = v_5'v_2'$. Jadi, dapat terlihat pula bahwa jika busur-busur di G bertetangga, peta dari busur-busur tersebut di G' juga bertetangga. Derajat setiap simpul pada kedua graf ini juga sama.

Definisi-definisi yang digunakan diatas merupakan definisi yang diambil dari (Rosen, 1995). Dalam skripsi ini, graf-graf yang akan dibahas adalah graf

berhingga, sederhana, dan tak berarah. Pada subbab berikutnya diberikan definisi dari beberapa kelas graf yang mengandung graf lintasan dan/atau graf lingkaran.

2.2 Jenis-jenis Graf

Graf lintasan (*path graph*), P_n , adalah graf dengan n simpul yang mempunyai busur-busur $v_1v_2, v_2v_3, \dots, v_{n-1}v_n$ atau dapat juga dinotasikan dalam $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_n$. Simpul v_1 disebut **simpul awal** dan v_n disebut **simpul akhir**. Semua simpul berderajat dua, kecuali untuk simpul awal dan simpul akhir berderajat satu. Pada graf lintasan, $|V| = n$ dan $|E| = n - 1$. **Graf lingkaran** (*cycle graph*), C_n , adalah graf yang diperoleh dari graf lintasan P_n yang diberi tambahan busur antara simpul awal dan simpul akhir, sehingga graf lingkaran merupakan graf teratur berderajat dua. Pada Gambar 2.3 (a) diberikan graf lintasan dengan 5 simpul (P_5) dan (b) diberikan graf lingkaran dengan 5 simpul (C_5).

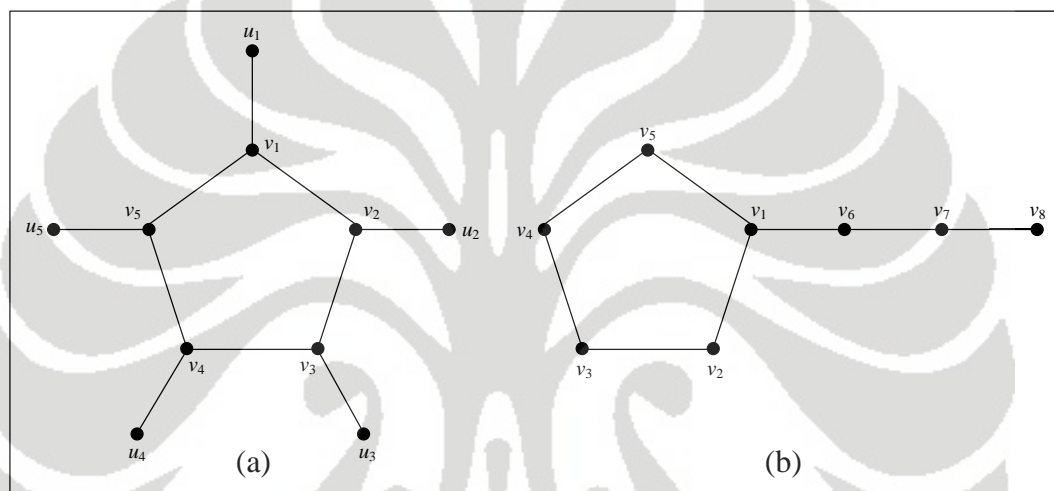


Gambar 2.3 (a) P_5 , (b) C_5

Graf matahari (*sun graph*), $C_n \odot \bar{K}_1$, adalah graf yang diperoleh dari graf lingkaran C_n dengan penambahan satu simpul berderajat satu di setiap simpul C_n . Simpul-simpul pada graf lingkaran dinotasikan dengan v_i yang disebut sebagai **simpul dalam** (*inner vertex*). Sedangkan simpul-simpul selain pada graf lingkaran dinotasikan dengan u_i yang disebut sebagai **simpul luar** (*outer vertex*). Secara matematis, graf matahari $C_n \odot \bar{K}_1$ memiliki himpunan simpul $V(C_n \odot \bar{K}_1) = \{v_i \mid 1 \leq i \leq n\} \cup \{u_i \mid 1 \leq i \leq n\}$ dan himpunan busur $E(C_n \odot \bar{K}_1) = \{v_i v_{i+1} \mid 1 \leq i \leq n\} \cup \{u_i v_i \mid 1 \leq i \leq n\}$ dimana $i+1$ merupakan modulo n . Nilai n menyatakan banyaknya

simpul pada graf lingkaran yang digunakan untuk membangun graf matahari se-hingga nilai n disebut sebagai ukuran dari graf matahari. Dalam graf matahari, $|V| = |E| = 2n$. Pada Gambar 2.4 (a) diberikan graf matahari berukuran 5 ($C_5 \odot \bar{K}_1$).

Graf kecebong (*tadpole graph*), $T_{m,n}$, adalah graf yang diperoleh dari graf lingkaran C_m dan graf lintasan P_n dengan penambahan satu busur yang menghubungkan salah satu simpul pada graf lingkaran dengan simpul awal atau simpul akhir pada graf lintasan. Dalam graf kecebong, $|V| = |E| = m+n$. Pada Gambar 2.4 (b) diberikan graf kecebong, $T_{5,3}$.



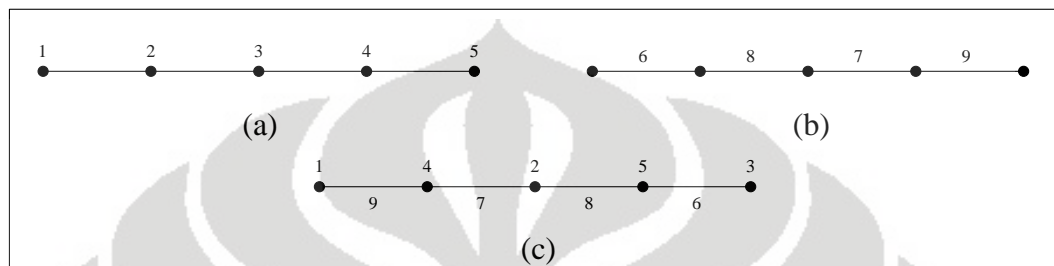
Gambar 2.4 (a) $C_5 \odot \bar{K}_1$, (b) $T_{5,3}$

Pada bab sebelumnya, telah disebutkan bahwa graf yang dibahas pada skripsi ini adalah graf lingkaran, graf matahari, dan graf kecebong. Kemudian pada subbab selanjutnya akan diberikan beberapa definisi mengenai pelabelan graf.

2.3 Pelabelan Graf

Pelabelan pada graf G adalah pemetaan bijektif λ dari himpunan simpul V atau himpunan busur E , atau keduanya ($V \cup E$) ke suatu himpunan, biasanya berupa himpunan bilangan bulat positif. Bilangan ini disebut sebagai **label**. Jika daerah asal dari pemetaan hanya himpunan simpul (himpunan busur), maka pelabelan disebut **pelabelan simpul (pelabelan busur)**. Jika daerah asal pemetaan

berupa gabungan dari himpunan simpul dan himpunan busur, maka pelabelan disebut **pelabelan total**. Pada Gambar 2.5 diberikan contoh graf P_5 dengan pelabelan simpul (a), pelabelan busur (b), dan pelabelan total (c). Untuk selanjutnya, pelabelan yang dibahas merupakan pelabelan total dengan daerah kawan berupa himpunan bilangan bulat positif berurutan yang dimulai dari 1.



Gambar 2.5 (a) Pelabelan simpul, (b) Pelabelan busur, (c) Pelabelan Total

Pada pelabelan total, jumlah dari semua label yang terkait dengan suatu elemen tertentu dari graf disebut **bobot**. Bobot dapat dihitung untuk simpul dan busur. **Bobot simpul** merupakan penjumlahan label simpul dengan semua label busur yang hadir pada simpul tersebut. **Bobot busur** merupakan penjumlahan label busur dengan label simpul-simpul yang dihubungkan dengan busur tersebut. Secara matematis, bobot simpul dari pelabelan λ dapat dinyatakan sebagai

$$w_{\lambda}(v) = \lambda(v) + \sum_{u \in N(v)} \lambda(uv)$$

dengan $N(v)$ adalah himpunan semua simpul yang bertetangga dengan v , $v \in V$. Sedangkan bobot busur dapat dinyatakan sebagai

$$w_{\lambda}(uv) = \lambda(u) + \lambda(uv) + \lambda(v), \forall uv \in E.$$

Suatu graf G dikatakan memiliki **pelabelan ajaib** apabila bobot untuk setiap simpul dan/atau busur bernilai k . Bilangan k disebut sebagai **konstanta ajaib**. Apabila $w_{\lambda}(v) = k$ untuk setiap $v \in V$, maka pelabelan disebut **pelabelan total simpul ajaib**. Apabila $w_{\lambda}(uv) = k$ untuk setiap $uv \in E$, maka pelabelan disebut **pelabelan total busur ajaib**.

Pada skripsi ini, pelabelan yang dibahas hanyalah mengenai pelabelan total simpul ajaib. Konsep pelabelan ini diperkenalkan oleh MacDougall *et al.* (2002) yang didefinisikan sebagai berikut:

Definisi 2.1 (MacDougall *et al.*, 2002)

Suatu pemetaan λ dari $V \cup E$ ke himpunan bilangan bulat $1, 2, \dots, |V|+|E|$ disebut **pelabelan total simpul ajaib** (PTSA) dari G jika λ merupakan pemetaan bijektif dengan sifat bahwa untuk setiap simpul $v \in V$,

$$w_\lambda(v) = \lambda(v) + \sum_{u \in N(v)} \lambda(uv) = k$$

dimana $N(v)$ adalah himpunan semua simpul yang bertetangga dengan v dan k merupakan konstanta ajaib untuk λ .

Pada PTSA didefinisikan pelabelan dual sebagai berikut:

Definisi 2.2 (MacDougall *et al.*, 2002)

Misalkan $\lambda: V \cup E \rightarrow \{1, 2, \dots, |V|+|E|\}$ adalah suatu PTSA pada graf G .

Definisikan $\lambda': V \cup E \rightarrow \{1, 2, \dots, |V|+|E|\}$ sebagai berikut

$$\lambda'(v) = |V| + |E| + 1 - \lambda(v)$$

untuk setiap $v \in V$, dan

$$\lambda'(uv) = |V| + |E| + 1 - \lambda(uv)$$

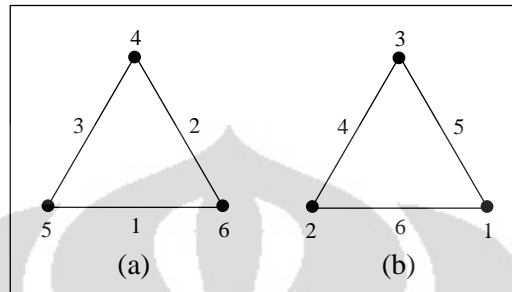
untuk setiap $uv \in E$.

Maka, λ' adalah **dual** dari λ .

Pada Gambar 2.6 diberikan 2 PTSA untuk graf lingkaran C_3 . Dapat terlihat bahwa jumlah semua bobot simpul untuk Gambar 2.6 (a) dan (b) adalah sama, yaitu masing-masing 9 dan 12. Dari gambar tersebut dapat terlihat pula bahwa label simpul dan busur untuk PTSA pada Gambar 2.6 (b) diperoleh berdasarkan rumus yang diberikan pada Definisi 2.2 sehingga PTSA pada Gambar 2.6 (b) merupakan dual dari PTSA pada Gambar 2.6 (a). Contohnya, perhatikan Gambar 2.6 (a). Misalkan $\lambda(v_1) = 4$ dan $\lambda(e_1) = 2$. Karena pada graf C_3 , $|V| = |E| = 3$, maka label simpul v_1 dan busur e_1 untuk dual dari PTSA pada Gambar 2.6 (a) masing-masing adalah $|V| + |E| + 1 - \lambda(v_1) = 3 + 3 + 1 - 4 = 3$ dan $|V| + |E| + 1 - \lambda(e_1) = 3 + 3 + 1 - 2 = 5$. Cara yang sama dilakukan untuk simpul dan busur berikutnya. Sehubungan dengan pelabelan dual ini, diberikan teorema berikut:

Teorema 2.1 (MacDougall *et al.*, 2002)

Dual dari suatu PTSA pada graf G juga merupakan suatu PTSA jika dan hanya jika G adalah graf teratur.



Gambar 2. 6 PTSA C_3 dengan $k = 9$ (a) dan dualnya dengan $k = 12$ (b)

Suatu graf G mungkin memiliki PTSA dengan beberapa nilai k yang berbeda. Batasan nilai k tersebut dapat diperoleh melalui perhitungan dasar (*basic counting*).

Misalkan S_v adalah jumlah label simpul dan S_e jumlah label busur G .

Karena label yang digunakan merupakan bilangan bulat $1, 2, \dots, |V|+|E|$, maka

$$S_v + S_e = \sum_{i=1}^{|V|+|E|} i. \quad (2.1)$$

Untuk setiap simpul v , $\lambda(v) + \sum_{u \in N(v)} \lambda(uv) = k$. Karena dua simpul yang bertetangga dihubungkan oleh satu busur, maka label busur berkontribusi dua kali untuk menghitung jumlah dari semua bobot simpul. Secara matematis,

$$S_v + 2S_e = |V|k. \quad (2.2)$$

Substitusi (2.1) ke (2.2),

$$S_e + \sum_{i=1}^{|V|+|E|} i = |V|k. \quad (2.3)$$

Setiap busur memiliki label yang berbeda (demikian juga untuk simpul). Busur dapat dilabel dengan label terkecil, label terbesar, ataupun label diantara label terkecil dan terbesar. Akibatnya,

$$\sum_{i=1}^{|E|} i \leq S_e \leq \sum_{i=|V|+1}^{|V|+|E|} i. \quad (2.4)$$

Substitusi (2.3) ke (2.4),

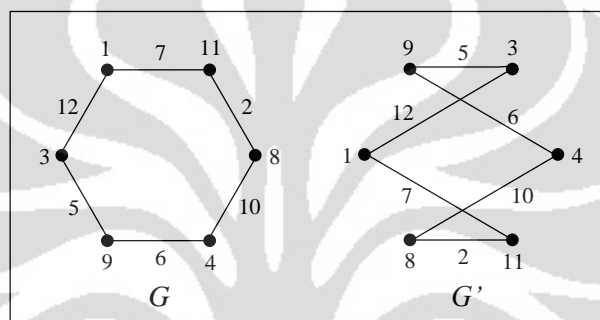
$$\sum_{i=1}^{|V|+|E|} i + \sum_{i=1}^{|E|} i \leq |V|k \leq 2 \sum_{i=1}^{|V|+|E|} i - \sum_{i=1}^{|V|} i. \quad (2.5)$$

Hasil ini akan memberikan batasan nilai k untuk PTSA pada graf G . Misalnya, diberikan contoh perhitungan batasan nilai k untuk PTSA pada graf matahari C_4

$\odot \bar{K}_1$. Pada graf $C_4 \odot \bar{K}_1$, nilai $|V| = |E| = 2(4) = 8$. Dengan substitusi nilai ini ke (2.5) akan diperoleh batasan nilai k sebagai berikut:

$$\begin{aligned} \sum_{i=1}^{16} i + \sum_{i=1}^8 i &\leq 8k \leq 2 \sum_{i=1}^{16} i - \sum_{i=1}^8 i \\ \frac{(16)(17)}{16} + \frac{(8)(9)}{16} &\leq k \leq 2 \left(\frac{(16)(17)}{16} \right) - \frac{(8)(9)}{16} \\ 21\frac{1}{2} &\leq k \leq 29\frac{1}{2}. \end{aligned}$$

Jadi, nilai k yang mungkin untuk graf $C_4 \odot \bar{K}_1$ adalah 22, 23, 24, 25, 26, 27, 28, 29.



Gambar 2.7 Dua PTSA yang isomorfik

Pada subbab sebelumnya telah dibahas mengenai dua graf tanpa label yang isomorfik. Namun, definisi graf isomorfik yang telah diberikan tersebut menjadi berbeda jika graf yang digunakan adalah graf dengan label. Dua graf $G = (V, E)$ dan $G' = (V', E')$ yang diberi label dikatakan **isomorfik** (memiliki **pelabelan isomorfik**) apabila terdapat pemetaan bijektif f dari V ke V' dengan sifat bahwa jika simpul-simpul pada V dan V' diberi label, maka simpul berlabel v_1 dan v_2 bertetangga di G jika dan hanya jika simpul berlabel $f(v_1)$ dan $f(v_2)$ bertetangga di G' . Hal ini berlaku untuk setiap simpul berlabel di G . Seperti pada definisi dua graf tanpa label yang isomorfik, jumlah simpul, jumlah busur, dan derajat dari setiap simpul pada kedua graf berlabel yang isomorfik dipertahankan sama. Pada Gambar 2.7 diberikan dua PTSA yang isomorfik pada graf G dan G' dengan nilai $k = 20$. Berdasarkan contoh tersebut dapat dilihat bahwa jika dua simpul berlabel bertetangga di G , maka peta dari kedua simpul berlabel tersebut juga bertetangga di G' . Demikian pula untuk busur. Jika dua busur berlabel bertetangga di G , maka peta dari kedua busur berlabel tersebut juga bertetangga di G' .

Hasil-hasil penelitian mengenai PTSA yang telah dipublikasikan lebih banyak membahas tentang eksistensi PTSA untuk kelas-kelas graf tertentu. Bukti dapat dilihat pada rujukan. Untuk graf lingkaran dengan banyak simpul n , (MacDougall *et al.*, 2002) telah membuktikan teorema berikut:

Teorema 2.2 (MacDougall *et al.*, 2002)

Graf lingkaran C_n memiliki suatu PTSA untuk nilai $n \geq 3$.

Untuk graf matahari $C_n \odot \bar{K}_1$, hasil penelitian yang telah dipublikasikan adalah mengenai eksistensi PTSA pada gabungan tak terhubung t graf matahari, baik gabungan t graf matahari yang isomorfik, maupun yang tidak isomorfik. Namun, yang menjadi bahasan pada skripsi ini hanya untuk nilai $t = 1$. Kedua hasil penelitian ini hanya berlaku untuk nilai k tertentu, bukan untuk semua nilai k yang mungkin berdasarkan perhitungan dasar. Untuk gabungan t graf matahari yang isomorfik, (Slamin *et al.*, 2006) telah membuktikan teorema berikut:

Teorema 2.3 (Slamin *et al.*, 2006)

Untuk $n \geq 3$ dan $t \geq 1$, gabungan t graf matahari $t(C_n \odot \bar{K}_1)$ memiliki suatu PTSA dengan nilai konstanta ajaib $k = 6nt + 1$.

Untuk gabungan t graf matahari yang tidak harus isomorfik, (Rahim & Slamin, 2008) telah membuktikan teorema berikut:

Teorema 2.4 (Rahim & Slamin, 2008)

Jika $t_i \geq 3$ untuk setiap $i = 1, 2, \dots, n$ dan $n \geq 1$, maka gabungan n graf matahari $(C_{t_1} \odot \bar{K}_1) \cup (C_{t_2} \odot \bar{K}_1) \cup \dots \cup (C_{t_n} \odot \bar{K}_1)$ memiliki suatu PTSA dengan nilai konstanta ajaib $k = 6 \sum_{k=1}^n t_k + 1$.

Hasil-hasil penelitian mengenai eksistensi PTSA untuk kelas-kelas graf yang lainnya dapat dilihat pada (Gallian, 2009).

Pendekatan lain yang dapat dilakukan untuk memperoleh semua PTSA yang mungkin dari suatu graf adalah menggunakan algoritma. Dalam makalah

(Baker & Sawada, 2008) telah dibahas algoritma untuk membangun PTSA yang tidak isomorfik pada graf lingkaran. Pada bab selanjutnya, akan diberikan ulasan ulang mengenai algoritma tersebut, kemudian dikembangkan untuk membangun algoritma-algoritma PTSA pada graf matahari dan graf kecebong. Sampai saat ini, belum ada hasil penelitian yang menyatakan bahwa graf kecebong memiliki suatu PTSA.



BAB 3

ALGORITMA PTSA UNTUK GRAF LINGKARAN, MATAHARI, DAN KECEBONG

Dalam Bab ini akan diberikan algoritma-algoritma PTSA untuk graf lingkaran, matahari, dan kecebong. Nilai-nilai k yang mungkin untuk PTSA pada masing-masing graf juga akan dibahas.

Pada Definisi 2.1 telah diberikan definisi dari Pelabelan Total Simpul Ajaib (PTSA). PTSA dari graf G merupakan suatu pemetaan bijektif dari $V \cup E$ ke himpunan bilangan bulat $1, 2, \dots, |V| + |E|$ dengan sifat bahwa bobot setiap simpul pada graf G bernilai sama, yaitu k . Pendekatan yang digunakan untuk membangun PTSA dalam skripsi ini adalah menggunakan algoritma yang bekerja secara iteratif untuk memperoleh label-label elemen pada graf. Untuk memperoleh algoritma yang efektif, PTSA yang isomorfik akan dihindari sehingga dapat diperoleh semua PTSA berbeda dari graf terkait.

Pada Bab sebelumnya telah dibahas mengenai pelabelan isomorfik. Dua graf G dan G' yang diberi label dikatakan memiliki pelabelan isomorfik apabila terdapat pemetaan bijektif dari label-label untuk simpul-simpul pada G ke label-label untuk simpul-simpul pada G' dengan sifat bahwa dua simpul berlabel pada G bertetangga jika dan hanya jika peta dari kedua simpul berlabel tersebut juga bertetangga di G' . Graf G dan G' yang dibahas selanjutnya adalah dua graf yang sama. Dari observasi dapat diketahui bahwa jika diberikan suatu graf dengan pelabelan tertentu, pelabelan yang isomorfik dari graf tersebut dapat diperoleh dengan melakukan rotasi atau refleksi terhadap label-labelnya. Pelabelan isomorfik yang merupakan hasil dari suatu rotasi disebut **kasus rotasional simetris** dan yang merupakan hasil dari suatu refleksi disebut **kasus refleksional simetris**. Kedua kasus ini dapat terjadi pada PTSA untuk graf lingkaran dan matahari. Sementara, untuk graf kecebong hanya kasus refleksional simetris yang dapat terjadi.

Pada (2.5) telah diberikan batasan nilai k untuk PTSA pada sembarang graf G . Untuk graf lingkaran C_n , nilai $|V| = |E| = n$. Dengan melakukan substitusi nilai ini ke (2.5) diperoleh

$$\begin{aligned}
\sum_{i=1}^{2n} i + \sum_{i=1}^n i &\leq nk \leq 2 \sum_{i=1}^{2n} i - \sum_{i=1}^n i \\
\frac{(2n)(2n+1)}{2n} + \frac{(n)(n+1)}{2n} &\leq k \leq 2 \left(\frac{(2n)(2n+1)}{2n} \right) - \frac{(n)(n+1)}{2n} \\
\frac{5n+3}{2} &\leq k \leq \frac{7n+3}{2}
\end{aligned} \tag{3.1}$$

Jadi, batasan nilai k untuk PTSA pada graf lingkaran C_n diberikan oleh (3.1).

Seperti yang telah disebutkan pada Bab 2, untuk graf matahari $C_n \odot \bar{K}_1$, $|V| = |E| = 2n$. Dengan melakukan substitusi nilai ini ke (2.5) diperoleh

$$\begin{aligned}
\sum_{i=1}^{4n} i + \sum_{i=1}^{2n} i &\leq 2nk \leq 2 \sum_{i=1}^{4n} i - \sum_{i=1}^{2n} i \\
\frac{(4n)(4n+1)}{4n} + \frac{(2n)(2n+1)}{4n} &\leq k \leq 2 \left(\frac{(4n)(4n+1)}{4n} \right) - \frac{(2n)(2n+1)}{4n} \\
5n + \frac{3}{2} &\leq k \leq 7n + \frac{3}{2}
\end{aligned} \tag{3.2}$$

Jadi, batasan nilai k untuk PTSA pada graf matahari $C_n \odot \bar{K}_1$ diberikan oleh (3.2).

Untuk graf kecebong $T_{m,n}$, $|V| = |E| = m+n$. Dengan melakukan substitusi nilai ini ke (2.5) diperoleh

$$\begin{aligned}
\sum_{i=1}^{2(m+n)} i + \sum_{i=1}^{m+n} i &\leq (m+n)k \leq 2 \sum_{i=1}^{2(m+n)} i - \sum_{i=1}^{m+n} i \\
\frac{(2(m+n))(2(m+n)+1)}{2(m+n)} + \frac{(m+n)(m+n+1)}{2(m+n)} &\leq k \leq 2 \left(\frac{(2(m+n))(2(m+n)+1)}{2(m+n)} \right) - \frac{(m+n)(m+n+1)}{2(m+n)} \\
\frac{5(m+n)+3}{2} &\leq k \leq \frac{7(m+n)+3}{2}
\end{aligned} \tag{3.3}$$

Jadi, batasan nilai k untuk PTSA pada graf kecebong $T_{m,n}$ diberikan oleh (3.3).

Batasan nilai k yang diberikan melalui perhitungan dasar di atas merupakan batasan yang diperoleh tanpa memperhatikan struktur khusus dari suatu graf G . Adanya simpul-simpul dengan derajat berbeda, seperti pada graf matahari dan kecebong, dapat memberikan batasan nilai k yang lebih ketat.

Untuk graf matahari $C_n \odot \bar{K}_1$, simpul-simpul luar u memiliki derajat satu dan simpul-simpul dalam v memiliki derajat tiga. Pertama, perhatikan penjumlahan dari semua bobot simpul dalam v . Batas bawah dari penjumlahan ini diperoleh dengan memberikan n label terkecil untuk busur dalam (karena label busur berkontribusi dua kali dalam penjumlahan semua bobot simpul dalam) dan $2n$ label terkecil berikutnya untuk simpul dalam dan busur luar. Kemudian, perhatikan penjumlahan dari semua bobot simpul luar u . Batas atas dari penjumlahan ini diperoleh dengan memberikan $2n$ label terbesar untuk simpul dan busur luar. Jadi,

$$\begin{aligned}
w(v_1) + w(v_2) + \dots + w(v_n) &= nk = w(u_1) + w(u_2) + \dots + w(u_n) \\
2 \sum_{i=1}^n i + \sum_{i=n+1}^{3n} i &\leq nk \leq \sum_{i=2n+1}^{4n} i
\end{aligned}$$

$$\frac{n(n+1)}{2n} + \frac{3n(3n+1)}{2n} \leq k \leq \frac{4n(4n+1)}{2n} - \frac{2n(2n+1)}{2n}$$

$$5n + 2 \leq k \leq 6n + 1 \quad (3.4)$$

Berdasarkan (3.4) ini diperoleh batasan nilai k untuk PTSA graf matahari $C_n \odot \bar{K}_1$ yang lebih ketat dibandingkan dengan (3.2).

Untuk graf kecebong $T_{m,n}$, simpul pada graf lingkaran C_m yang memiliki satu busur tambahan, yaitu simpul v_1 , memiliki derajat tiga, sedangkan simpul-simpul lainnya berderajat dua dan satu. Maka, batas bawah nilai konstanta ajaib k tidak mungkin lebih kecil dari nilai minimum dari bobot simpul berderajat tiga, yaitu

$$k \geq \sum_{i=1}^4 i = \frac{4(5)}{2} = 10 \quad (3.5)$$

Sekarang, perhatikan penjumlahan semua bobot simpul berderajat dua dengan simpul berderajat satu. Batas atas penjumlahan ini diperoleh dengan memberikan $m+n-3$ label terbesar untuk busur-busurnya, kecuali tiga busur yang hadir pada simpul v_1 , dan $m+n+2$ label terbesar berikutnya untuk simpul-simpulnya dan tiga busur yang hadir pada simpul v_1 . Jadi,

$$(m+n-1)k = w(v_2) + \dots + w(v_m) + w(v_{m+1}) \dots + w(v_{m+n})$$

$$(m+n-1)k \leq 2 \sum_{i=m+n+4}^{2(m+n)} i + \sum_{i=2}^{m+n+3} i$$

$$= 2 \sum_{i=1}^{2(m+n)} i - \sum_{i=1}^{m+n+3} i - 1$$

$$= 2 \left(\frac{(2m+2n)(2m+2n+1)}{2} \right) - \frac{(m+n+3)(m+n+4)}{2} - 1$$

$$k \leq \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)} \quad (3.6)$$

Jadi, berdasarkan (3.5) dan (3.6) diperoleh

$$10 \leq k \leq \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)} \quad (3.7)$$

Maka, batasan nilai k yang lebih ketat untuk PTSA graf kecebong $T_{m,n}$ merupakan irisan dari (3.3) dan (3.7), yaitu

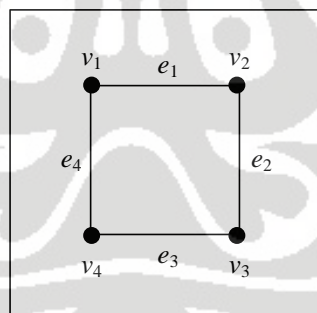
$$\max \left(\frac{5(m+n)+3}{2}, 10 \right) \leq k \leq \min \left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)} \right) \quad (3.8)$$

Pada dasarnya, algoritma pelabelan sembarang graf secara umum bersifat *NP-complete*. Oleh karena itu, pembangunan algoritma pelabelan biasanya dilakukan untuk kelas-kelas graf tertentu. Misalnya, Baker dan Sawada (2008) telah membangun algoritma PTSA untuk graf lingkaran C_n dan graf roda W_n . Pada

Subbab 3.1 akan dibahas mengenai pembangunan algoritma PTSA untuk graf lingkaran C_n versi Baker dan Sawada. Kemudian, algoritma ini dikembangkan untuk membangun algoritma PTSA untuk graf matahari $C_n \odot \bar{K}_1$ yang akan diberikan pada Subbab 3.2, dan untuk graf kecebong $T_{m,n}$ yang akan diberikan pada Subbab 3.3.

3.1 Algoritma PTSA untuk Graf Lingkaran

Algoritma PTSA untuk graf lingkaran C_n diberikan oleh Baker dan Sawada (2008). Yang menjadi masukan (*input*) dalam algoritma PTSA ini adalah n dan k , yang menyatakan banyaknya simpul pada graf lingkaran dan konstanta ajaib yang mungkin, yang bergantung pada nilai n , seperti yang diberikan pada (3.1). Karena PTSA merupakan pemetaan bijektif dari $V \cup E$ ke $\{1, 2, \dots, |V| + |E|\}$ dan pada graf lingkaran nilai $|V| = |E| = n$, maka himpunan label yang tersedia untuk melabel elemen-elemen pada graf C_n adalah $\{1, 2, \dots, 2n\}$. Setiap label ini harus digunakan tepat satu kali.



Gambar 3.1 Graf C_4

Pada Bab sebelumnya telah diberikan definisi dari graf lingkaran C_n beserta contohnya. Simpul-simpul pada graf tersebut ditulis dengan v_1, v_2, \dots, v_n dan busur-busurnya ditulis sebagai pasangan kedua simpul ujungnya, yaitu $v_1v_2, v_2v_3, \dots, v_{n-1}v_n$. Akan tetapi, pada algoritma PTSA yang akan diberikan, busur-busur pada graf lingkaran tidak ditulis sebagai pasangan kedua simpul ujungnya, melainkan ditulis sebagai e . Misalnya, busur e_1 adalah busur yang diperoleh dari pasangan simpul ujung v_1v_2 , busur e_2 adalah busur yang diperoleh dari pasangan

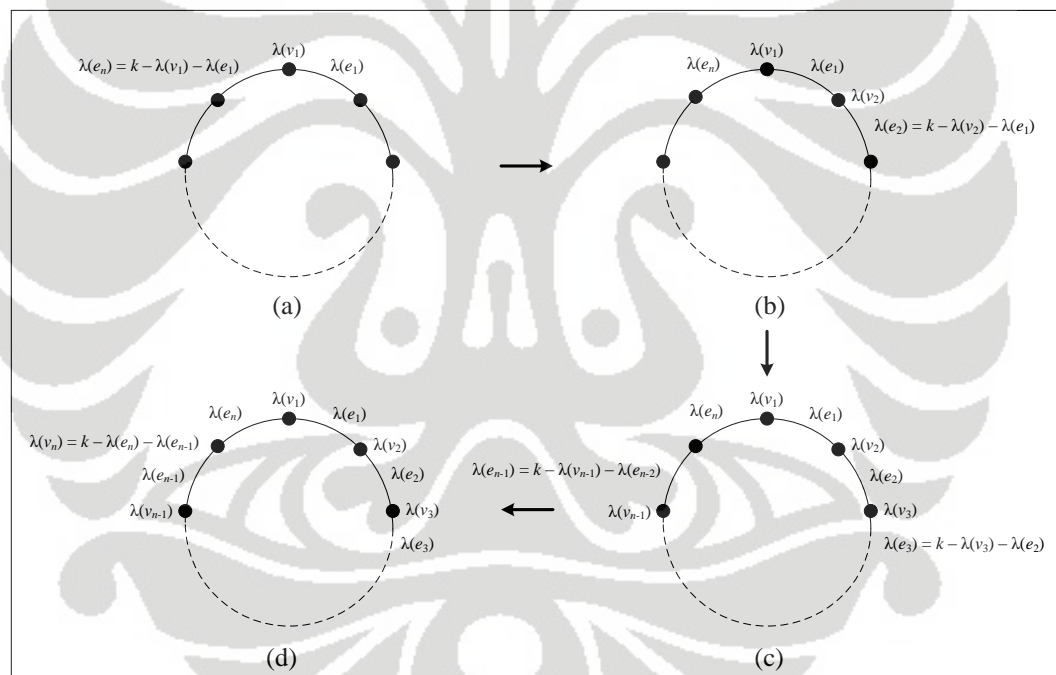
simpul ujung v_2v_3 , dan seterusnya. Sedangkan untuk simpul tetap ditulis sebagai v . Pada Gambar 3.1 diberikan contoh penamaan untuk graf lingkaran C_4 .

Dilihat dari struktur graf lingkaran, selalu terdapat dua busur yang hadir pada setiap simpul. Jadi, untuk menghitung bobot simpul dari graf lingkaran yang memiliki suatu pelabelan total simpul, label yang berkontribusi adalah label simpul yang berkaitan dan label dua busur yang hadir pada simpul tersebut. Secara umum, algoritma PTSA pada graf lingkaran ini dimulai dengan mendapatkan satu label simpul dan satu label busur yang hadir pada simpul tersebut. Kemudian, label busur lain yang hadir pada simpul tersebut ditentukan dengan cara menghitung selisih dari nilai konstanta ajaib k yang dimasukkan sebagai *input* dengan label untuk simpul dan busur yang telah didapatkan. Label yang diperoleh dengan cara ini disebut sebagai *determined label*. Pada tahap akhir proses pelabelan, akan terdapat satu simpul yang belum dilabel. Label untuk simpul ini dapat ditentukan dengan cara menghitung selisih dari nilai k dengan kedua label busur yang hadir pada simpul tersebut. Label ini juga disebut sebagai *determined label*. Jadi, *determined label* bisa merupakan label busur, maupun label simpul. *Determined label* ini mungkin tidak tersedia. Jika hal ini terjadi, maka algoritma PTSA tidak dapat dilanjutkan dan harus melakukan *backtracking* (penelusuran langkah yang dilakukan secara mundur untuk mengganti label yang digunakan). Kondisi-kondisi yang menyebabkan *determined label* $\lambda(x)$ tidak tersedia adalah:

1. $\lambda(x) < 1$, karena label untuk membentuk suatu PTSA merupakan bilangan bulat positif,
2. $\lambda(x) > 2n$, karena $|V| + |E| = 2n$,
3. $\lambda(x)$ telah digunakan untuk melabel elemen pada graf lingkaran.

Pada Gambar 3.2 diberikan ilustrasi pembentukan PTSA untuk graf lingkaran C_n dengan konstanta ajaib k . Pertama-tama, diberikan label untuk simpul v_1 dan busur e_1 , sebut dengan $\lambda(v_1)$ dan $\lambda(e_1)$ (Gambar 3.2 (a)). Karena simpul v_1 dan busur e_1 sudah diberi label, maka label untuk busur e_4 dapat ditentukan, yaitu $\lambda(e_4) = k - \lambda(v_1) - \lambda(e_1)$. Label busur e_4 ini merupakan *determined label*. *Determined label* ini bisa tersedia atau tidak tersedia. Pada ilustrasi ini, diasumsikan semua *determined label* tersedia. Selanjutnya, diberikan label untuk simpul v_2 , yaitu $\lambda(v_2)$. Karena simpul v_2 dan busur e_1 sudah diberi label, maka label untuk

busur e_2 dapat ditentukan, yaitu $\lambda(e_2) = k - \lambda(v_2) - \lambda(e_1)$ (Gambar 3.2 (b)). Label busur e_2 ini merupakan *determined label*. Kemudian, diberikan label untuk simpul v_3 , yaitu $\lambda(v_3)$. Karena simpul v_3 dan busur e_2 sudah diberi label, maka label untuk busur e_3 dapat ditentukan, yaitu $\lambda(e_3) = k - \lambda(v_3) - \lambda(e_2)$. Label busur e_3 ini merupakan *determined label*. Proses yang sama dilakukan sampai melabel simpul v_{n-1} dan kemudian menentukan label busur e_{n-1} (Gambar 3.2 (c)). Langkah berikutnya adalah diberikan label untuk satu-satunya elemen pada graf C_n yang belum dilabel, yaitu simpul v_n . Label ini merupakan *determined label*, yaitu $\lambda(v_n) = k - \lambda(e_n) - \lambda(e_{n-1})$ (Gambar 3.2 (d)). Dari Gambar 3.2 (b) dan (c) dapat terlihat bahwa terdapat proses yang sama dalam melabel elemen graf C_n , yaitu melabel simpul kemudian label busur dapat ditentukan.



Gambar 3.2 Proses pembentukan PTSA pada graf C_n

Berdasarkan adanya kesamaan proses melabel elemen pada graf lingkaran, algoritma PTSA untuk graf lingkaran terbagi menjadi dua fungsi, yaitu fungsi inisialisasi (*initializeCycle*) dan fungsi perluasan (*extendCycle*). Pada fungsi inisialisasi, pertama-tama dilabel simpul v_1 dengan label yang tersedia, sebut dengan $\lambda(v_1)$. Kemudian, tandai $\lambda(v_1)$ agar label ini tidak dapat digunakan lagi. Elemen berikutnya yang dilabel adalah busur e_1 , sebut dengan $\lambda(e_1)$, dan tandai

$\lambda(e_1)$ agar label ini tidak dapat digunakan lagi. Label untuk busur e_n merupakan *determined label* dengan nilai $\lambda(e_n) = k - \lambda(v_1) - \lambda(e_1)$. Jika $\lambda(e_n)$ masih tersedia, maka tandai $\lambda(e_n)$ agar tidak dapat digunakan lagi dan akan dipanggil fungsi perluasan untuk melabel simpul dan busur berikutnya. Namun, jika $\lambda(e_n)$ tidak tersedia karena memenuhi salah satu dari tiga kondisi pada *determined label* yang mengakibatkan algoritma tidak dapat dilanjutkan, maka dilakukan *backtracking* untuk mengganti label dari elemen yang dilabel sebelum busur e_n , yaitu busur e_1 . Sebelum mengganti label busur e_1 ini, label busur e_1 yang digunakan sebelumnya harus dibuat kembali tersedia sehingga dapat digunakan lagi untuk melabel elemen yang lainnya. Jika ternyata tidak ada lagi label yang mungkin untuk melabel busur e_1 , maka kembali lakukan *backtracking* untuk mengganti label dari elemen yang dilabel sebelum busur e_1 , yaitu simpul v_1 . Sebelum mengganti label simpul v_1 ini, kembalikan dahulu label yang digunakan sebelumnya untuk melabel simpul v_1 sehingga label tersebut menjadi tersedia.

Fungsi perluasan adalah fungsi yang dibuat berdasarkan adanya kesamaan proses melabel elemen pada graf lingkaran, yaitu mencoba label yang masih tersedia untuk melabel simpul berikutnya dan kemudian menentukan label busur berikutnya dengan melakukan perhitungan karena label busur ini merupakan *determined label*. Pada fungsi ini digunakan parameter t sebagai posisi dari simpul yang akan dilabel. Fungsi inisialisasi akan memanggil fungsi perluasan dengan parameter $t = 2$. Artinya, simpul yang akan dilabel berikutnya adalah simpul v_2 , misalkan dilabel dengan $\lambda(v_2)$, dan tandai $\lambda(v_2)$ agar label ini tidak dapat digunakan lagi. Setelah itu, label untuk busur e_2 dapat ditentukan, yaitu $\lambda(e_2) = k - \lambda(v_2) - \lambda(e_1)$. Jika $\lambda(e_2)$ masih tersedia, maka tandai $\lambda(e_2)$ agar label ini tidak dapat digunakan lagi dan fungsi perluasan akan memanggil dirinya sendiri dengan parameter $t+1$. Namun, jika $\lambda(e_2)$ tidak tersedia, maka dilakukan *backtracking* untuk mengganti label dari elemen yang dilabel sebelum busur e_2 , yaitu simpul v_2 . Jika tidak ada lagi label yang mungkin untuk melabel simpul v_2 , maka kembali dilakukan *backtracking* ke fungsi inisialisasi untuk mengganti label dari elemen yang dilabel sebelum simpul v_2 , yaitu busur e_n . Karena label busur e_n merupakan *determined label*, maka lakukan *backtracking* untuk mengganti label dari elemen yang dilabel sebelum elemen dengan *determined label* tersebut, yaitu busur e_1 . Jika ti-

dak ada lagi label yang mungkin untuk melabel busur e_1 , maka *backtracking* dilakukan lagi untuk mengganti label dari elemen yang dilabel sebelum busur e_1 , yaitu simpul v_1 . Setiap kali dilakukan penggantian label suatu elemen pada proses *backtracking*, ubah dahulu status label yang digunakan sebelumnya untuk melabel elemen tersebut sehingga menjadi tersedia kembali.

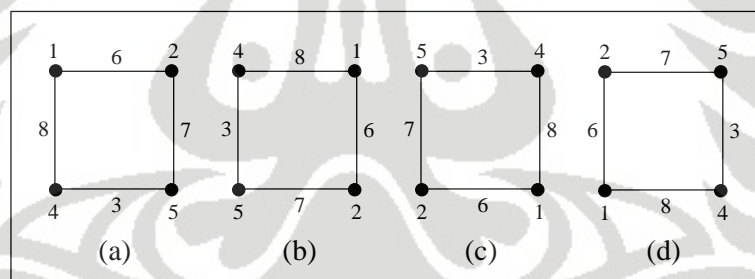
Proses rekursif pada fungsi perluasan akan berhenti apabila nilai $t = n$. Pada saat nilai $t = n$ akan terdapat satu simpul yang belum dilabel, yaitu simpul v_n . Label untuk simpul v_n ini merupakan *determined label*, yaitu $\lambda(v_n) = k - \lambda(e_n) - \lambda(e_{n-1})$. Jika $\lambda(v_n)$ merupakan satu-satunya label yang tersedia, maka telah terbentuk suatu PTSA untuk graf lingkaran C_n sehingga label akan dicetak. Jika $\lambda(v_n)$ tidak tersedia, lakukan *backtracking*. Setelah diperoleh suatu PTSA, proses tidak langsung berhenti. Komputasi tetap dilanjutkan secara *backtracking* sampai dihasilkan semua PTSA C_n dengan konstanta ajaib k .

<p>Algoritma 1.a. Fungsi initializeCycle()</p> <pre> for each available label i do $\lambda(v_1) := i$ $avail[i] := false$ for each available label j do $\lambda(e_1) := j$ $avail[j] := false$ $\lambda(e_n) := k - \lambda(v_1) - \lambda(e_1)$ if $0 < \lambda(e_n) \leq 2n$ and $avail[\lambda(e_n)]$ then $avail[\lambda(e_n)] := false$ extendCycle(2) $avail[\lambda(e_n)] := true$ end if $avail[j] := true$ end for $avail[i] := true$ end for </pre>	<p>Algoritma 1.b. Fungsi extendCycle(t)</p> <pre> if $t = n$ then $\lambda(v_n) := k - \lambda(e_n) - \lambda(e_{n-1})$ if $0 < \lambda(v_n) \leq 2n$ and $avail[\lambda(v_n)]$ then print() end if else for each available label i do $\lambda(v_t) := i$ $avail[i] := false$ $\lambda(e_t) := k - \lambda(v_t) - \lambda(e_{t-1})$ if $0 < \lambda(e_t) \leq 2n$ and $avail[\lambda(e_t)]$ then $avail[\lambda(e_t)] := false$ extendCycle($t+1$) $avail[\lambda(e_t)] := true$ end if $avail[i] := true$ end for end if </pre>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Algoritma PTSA untuk graf lingkaran diberikan pada Algoritma 1, dimana Algoritma 1.a. merupakan fungsi inisialisasi dan Algoritma 1.b. merupakan fungsi perluasan. Pada algoritma-algoritma ini, himpunan label yang tersedia mula-mula untuk melabel elemen pada suatu graf dinyatakan tersedia, ditandai dengan $avail[\lambda(x)]$ bernilai *true*. *Avail* ini merupakan suatu *array* dengan ukuran $2n$ dan

$\lambda(x)$ merupakan label yang digunakan untuk melabel elemen x pada graf. Nilai *false* akan diberikan pada *avail* jika $\lambda(x)$ telah digunakan untuk melabel elemen x , sehingga label tersebut tidak tersedia lagi. Dapat diperhatikan bahwa pada algoritma-algoritma yang diberikan terjadi pemanggilan fungsi secara rekursif.

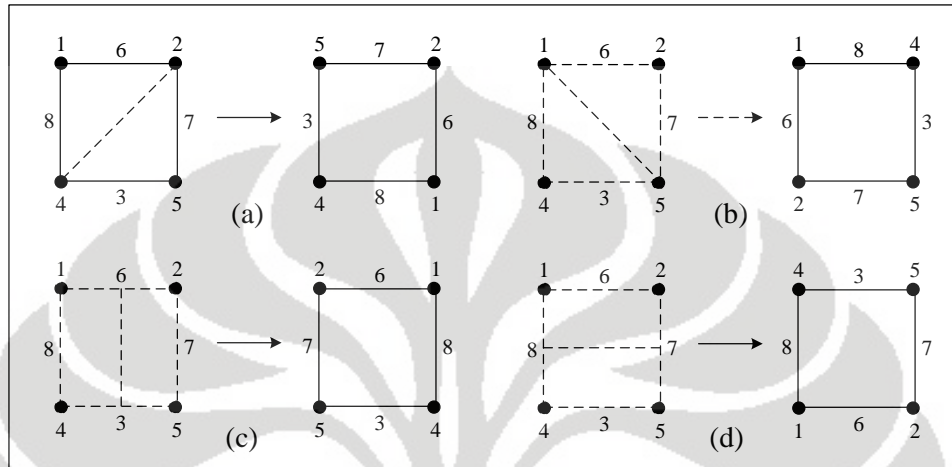
Algoritma 1 dapat diefektifkan dengan menghindari pelabelan-pelabelan yang isomorfik, sehingga yang dihasilkan adalah semua PTSA tidak isomorfik pada graf lingkaran yang diberikan. Pada awal Bab ini telah disebutkan bahwa pelabelan yang isomorfik dapat dihasilkan dengan melakukan rotasi atau refleksi terhadap suatu pelabelan yang ada. Pada Gambar 3.3 diberikan semua PTSA isomorfik untuk graf C_4 dengan $k = 15$ yang merupakan kasus rotasional simetris. Dari gambar ini dapat terlihat bahwa untuk menghindari PTSA isomorfik dengan kasus ini, salah satu simpul harus dilabel dengan label yang lebih kecil dari label simpul-simpul lainnya. Dalam hal ini, simpul v_1 dipilih sebagai simpul dengan label minimum. Syarat ini diberikan pada fungsi perluasan karena fungsi perluasan menentukan label untuk simpul-simpul lain selain simpul v_1 . Dengan menggunakan syarat ini, PTSA pada Gambar 3.3 (b), (c), dan (d) tidak mungkin terjadi.



Gambar 3.3 PTSA C_4 yang isomorfik dengan kasus rotasional simetris

Pada Gambar 3.4 diberikan semua PTSA isomorfik untuk graf C_4 dengan $k = 15$ yang merupakan kasus refleksional simetris dengan sumbu simetri yang dinyatakan oleh garis putus-putus. Dari gambar ini dapat terlihat bahwa salah satu cara untuk menghindari PTSA hasil refleksi pada Gambar 3.4 (a) dan (b) adalah dengan memberikan syarat pada dua busur berlabel yang bertetangga, yaitu salah satu label busurnya lebih kecil dari label busur yang satu. Dalam hal ini, dipilih label untuk busur e_1 lebih kecil dari label untuk busur e_n ($\lambda(e_1) < \lambda(e_n)$). Syarat ini diberikan pada fungsi inisialisasi karena fungsi inisialisasi menentukan label un-

tuk e_1 dan e_n . Karena pada fungsi inialisasi diberikan syarat bahwa $\lambda(e_1) < \lambda(e_n)$ dan pada fungsi perluasan diberikan syarat bahwa label simpul v_1 lebih kecil dari label simpul lainnya, maka PTSA hasil refleksi pada Gambar 3.4 (c) dan (d) tidak mungkin terjadi.

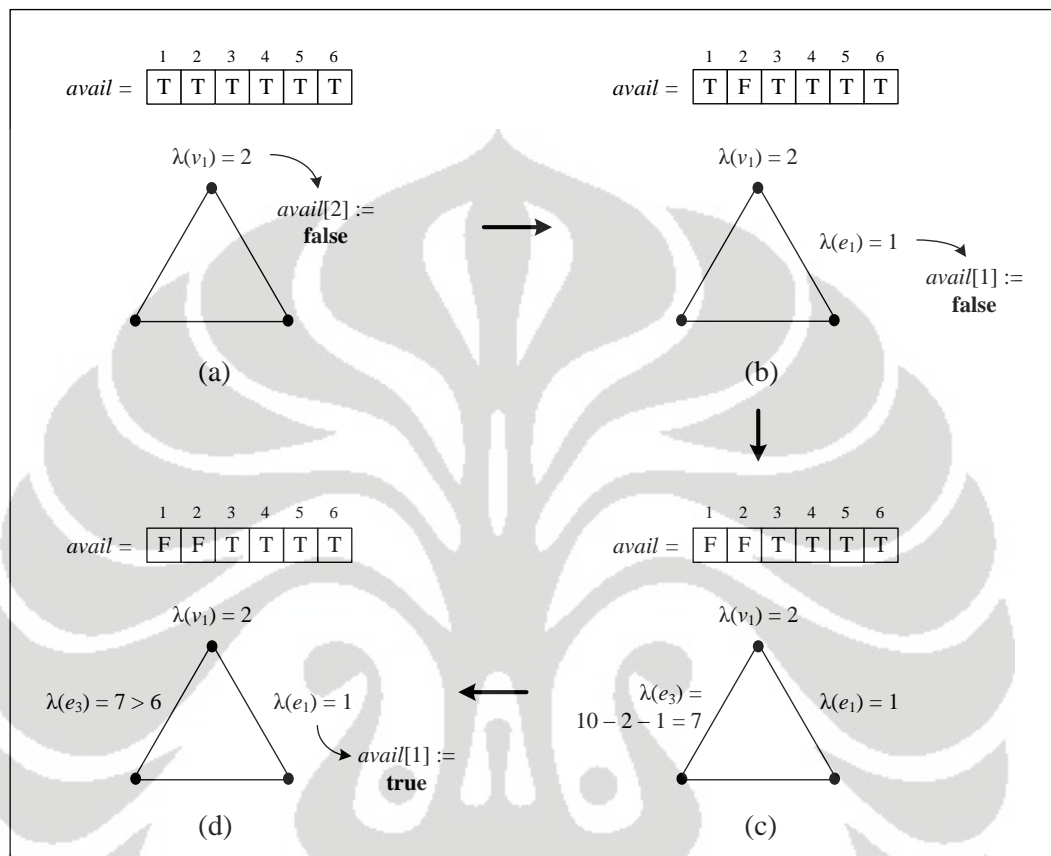


Gambar 3.4 PTSA C_4 yang isomorfik dengan kasus refleksional simetris

<p>Algoritma 2.a. Fungsi initializeCycle()</p> <pre> 1 for each available label i where $i \leq n+1$ do 2 $\lambda(v_1) := i$ 3 $avail[i] := false$ 4 for each available label j do 5 $\lambda(e_1) := j$ 6 $avail[j] := false$ 7 $\lambda(e_n) := k - \lambda(v_1) - \lambda(e_1)$ 8 if $\lambda(e_1) < \lambda(e_n) \leq 2n$ and $avail[\lambda(e_n)]$ then 9 $avail[\lambda(e_n)] := false$ 10 extendCycle(2) 11 $avail[\lambda(e_n)] := true$ 12 end if 13 end for 14 $avail[i] := true$ 15 end for </pre>	<p>Algoritma 2.b. Fungsi extendCycle(t)</p> <pre> 1 if $t = n$ then 2 $\lambda(v_n) := k - \lambda(e_n) - \lambda(e_{n-1})$ 3 if $\lambda(v_1) < \lambda(v_n) \leq 2n$ and $avail[\lambda(v_n)]$ then 4 print() 5 end if 6 else 7 for each available label i where $i > \lambda(v_1)$ do 8 $\lambda(v_t) := i$ 9 $avail[i] := false$ 10 $\lambda(e_t) := k - \lambda(v_t) - \lambda(e_{t-1})$ 11 if $0 < \lambda(e_t) \leq 2n$ and $avail[\lambda(e_t)]$ then 12 $avail[\lambda(e_t)] := false$ 13 extendCycle($t+1$) 14 $avail[\lambda(e_t)] := true$ 15 end if 16 end for 17 $avail[i] := true$ 18 end if </pre>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Oleh karena simpul v_1 merupakan simpul dengan label minimum, maka iterasi pada fungsi inialisasi dapat dipercepat dengan menambahkan syarat bahwa label maksimum untuk simpul v_1 adalah $n+1$. Apabila simpul v_1 merupakan simpul yang dilabel dengan label $n+1$, maka simpul-simpul lainnya dapat dilabel

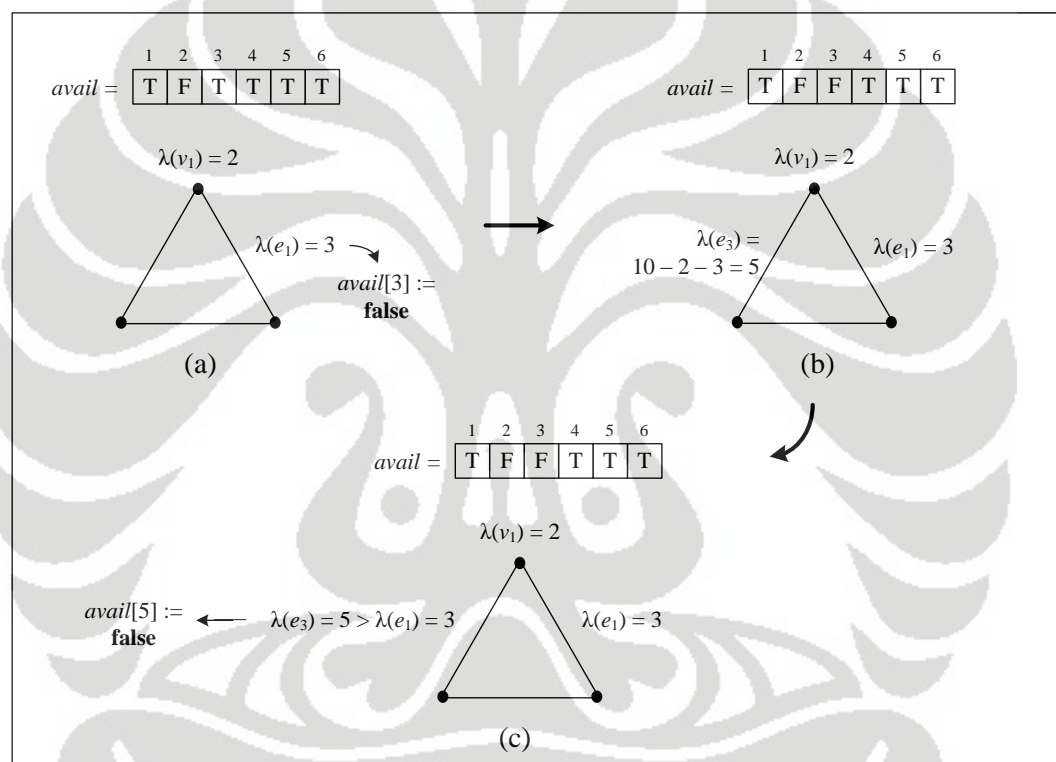
dengan $n - 1$ label tersisa yang lebih besar dari $n+1$. Algoritma 2.a. dan 2.b. masing-masing memberikan algoritma untuk fungsi inisialisasi dan fungsi perluasan pada graf lingkaran yang telah diefektifkan.



Gambar 3.5 Proses pada *initializeCycle* untuk membentuk PTSA C_3 dengan $k = 10$

Sebagai contoh penggunaan dari algoritma PTSA ini akan dilakukan pelabelan untuk graf C_3 dengan nilai $k = 10$. Label-label yang digunakan untuk membentuk PTSA C_3 adalah 1 sampai $2n = 2(3) = 6$, yaitu 1, 2, 3, 4, 5, 6, sehingga $avail = [T, T, T, T, T, T]$. Menggunakan fungsi inisialisasi yang diberikan pada Algoritma 2.a., langkah pertama yang dilakukan adalah mencoba label yang tersedia untuk melabel simpul v_1 . Misalkan pilih 2, maka $\lambda(v_1) = 2$ dan nyatakan $avail[2] = false$ (Gambar 3.5 (a)). Artinya, label 2 sudah digunakan untuk melabel simpul v_1 , sehingga $avail = [T, F, T, T, T, T]$. Langkah selanjutnya adalah mencoba label yang tersedia untuk melabel busur e_1 . Misalkan pilih 1, maka $\lambda(e_1) = 1$ dan nyatakan $avail[1] = false$ (Gambar 3.5 (b)). Ini berarti, label 1 sudah diguna-

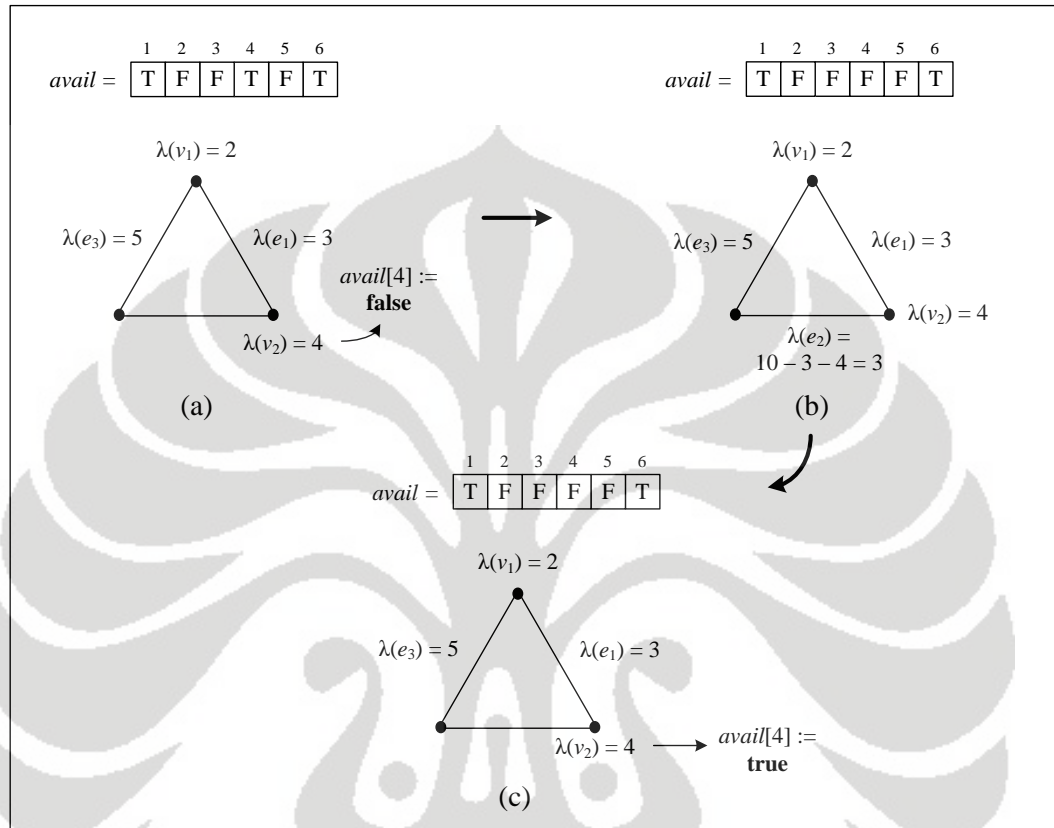
kan untuk melabel busur e_1 dan $avail = [F, F, T, T, T, T]$. Label untuk busur e_3 dapat ditentukan dengan menghitung $\lambda(e_3) = k - \lambda(v_1) - \lambda(e_1) = 7$ (Gambar 3.5 (c)). Label ini tidak tersedia karena $\lambda(e_3) = 7 > 2(3) = 6$. Karena label 7 tidak tersedia, maka proses berikutnya adalah menuju baris 12 pada Algoritma 2.a., yaitu melakukan *backtracking*. Nyatakan $avail[1] = true$ (Gambar 3.5 (d)), yang berarti label 1 tersedia kembali, sehingga $avail = [T, F, T, T, T, T]$. Label 1 ini merupakan label dari elemen yang dilabel sebelum melabel busur e_3 , yaitu busur e_1 . Dengan kata lain, busur e_1 gagal dilabel oleh 1.



Gambar 3.6 Backtracking pada *initializeCycle*

Langkah selanjutnya adalah kembali ke baris 4 pada Algoritma 2.a., yaitu busur e_1 akan dilabel kembali dengan label lain yang masih tersedia. Misalkan pilih 3, maka $\lambda(e_1) = 3$ dan nyatakan $avail[3] = false$ (Gambar 3.6 (a)). Artinya, label 3 sudah digunakan untuk melabel busur e_1 dan $avail = [T, F, F, T, T, T]$. Label untuk busur e_3 dapat ditentukan dengan menghitung $\lambda(e_3) = k - \lambda(v_1) - \lambda(e_1) = 5$ (Gambar 3.6 (b)). Karena $\lambda(e_1) = 3 < \lambda(e_3) = 5$ dan label 5 masih tersedia, maka label ini diterima untuk melabel busur e_3 dan nyatakan $avail[5] = false$ (Gambar

3.6 (c)). Ini berarti, label 5 sudah digunakan untuk melabel busur e_3 , sehingga $avail = [T, F, F, T, F, T]$. Kemudian, fungsi inialisasi ini akan memanggil fungsi $extendCycle$ dengan parameter $t = 2$.

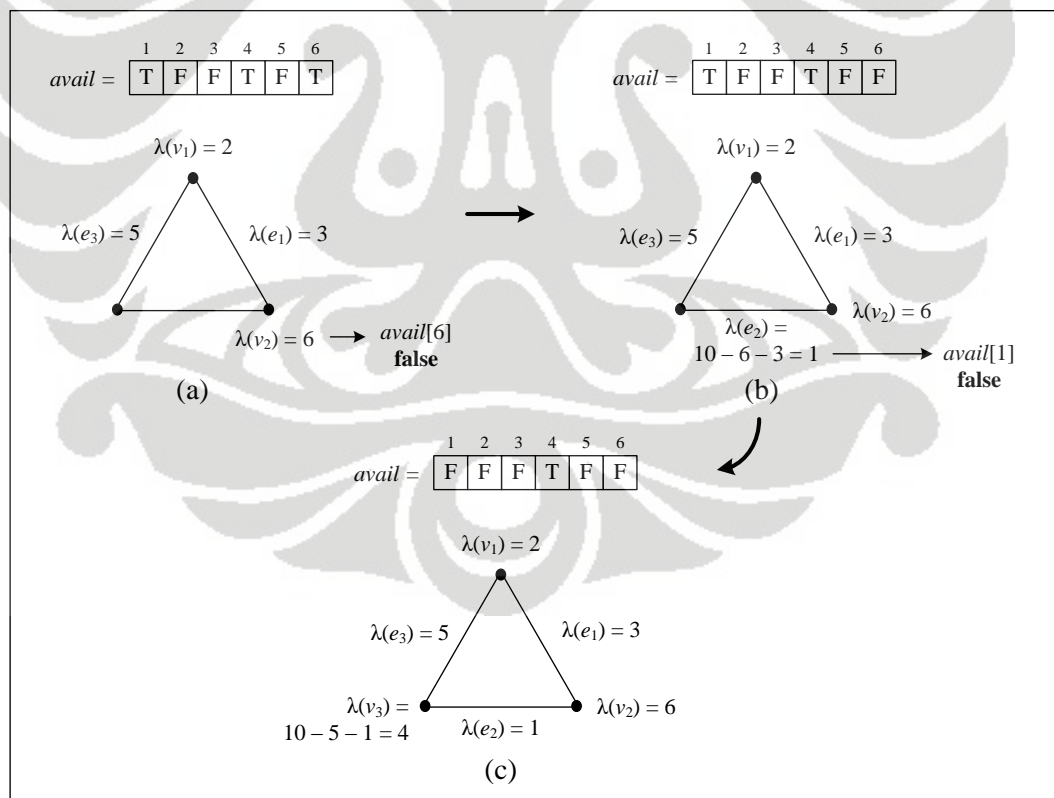


Gambar 3.7 Proses pada $extendCycle(2)$ untuk membentuk PTSA C_3 dengan $k = 10$

Parameter 2 pada fungsi $extendCycle(2)$ menyatakan bahwa elemen graf C_3 yang dilabel berikutnya adalah simpul ke-2. Karena simpul ini bukan simpul terakhir, maka proses berlanjut ke baris 5 pada Algoritma 2.b. Label yang dipilih untuk melabel simpul v_2 adalah yang lebih besar dari $\lambda(v_1) = 2$. Misalkan dipilih 4 sebagai label untuk simpul ini, maka $\lambda(v_2) = 4$ dan nyatakan $avail[4] = false$ (Gambar 3.7 (a)). Ini berarti, label 4 sudah digunakan untuk melabel simpul v_2 , sehingga $avail = [T, F, F, F, F, T]$. Label untuk busur e_2 dapat ditentukan dengan menghitung $\lambda(e_2) = k - \lambda(v_2) - \lambda(e_1) = 3$ (Gambar 3.7 (b)). Label 3 ternyata sudah tidak tersedia karena telah digunakan sebagai label busur e_1 , maka proses selanjutnya adalah menuju baris ke-14 pada Algoritma 2.b. untuk melakukan *backtrack*-

ing. Nyatakan $avail[4] = true$ yang berarti label 4 tersedia kembali (Gambar 3.7 (c)), sehingga $avail = [T, F, F, T, F, T]$. Label 4 ini merupakan label untuk elemen yang dilabel sebelum melabel busur e_2 , yaitu simpul v_2 . Dengan kata lain, simpul v_2 gagal dilabel oleh 4.

Proses selanjutnya adalah menuju baris ke-6 pada Algoritma 2.b. untuk melabel kembali simpul v_2 dengan label lain yang lebih besar dari $\lambda(v_1) = 2$. Satu-satunya label yang mungkin untuk melabel simpul ini adalah 6, maka $\lambda(v_2) = 6$ dan nyatakan $avail[6] = false$ (Gambar 3.8 (a)). Artinya, label 6 telah digunakan untuk melabel simpul v_2 , sehingga $avail = [T, F, F, T, F, F]$. Label untuk busur e_2 dapat ditentukan, yaitu $\lambda(e_2) = k - \lambda(v_2) - \lambda(e_1) = 1$. Karena label ini masih tersedia, maka nyatakan $avail[1] = false$ (Gambar 3.8 (b)). Ini berarti label 1 telah digunakan sebagai label busur e_2 , sehingga $avail = [F, F, F, T, F, F]$. Kemudian, fungsi $extendCycle$ ini akan memanggil dirinya sendiri dengan parameter $t+1 = 2+1 = 3$.



Gambar 3.8 Backtracking pada $extendCycle(2)$ dan $extendCycle(3)$ diproses untuk membentuk PTSA C_3 dengan $k = 10$

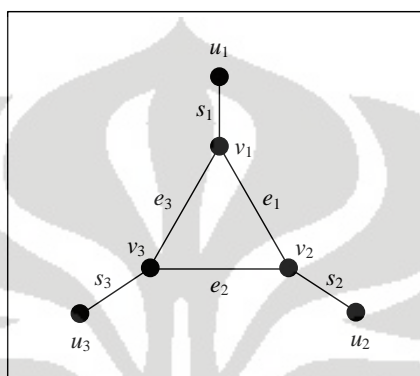
Parameter 3 pada fungsi $extendCycle(3)$ menyatakan bahwa elemen graf C_3 yang dilabel berikutnya adalah simpul ke-3. Karena simpul ini merupakan simpul terakhir pada graf C_3 , maka proses berlanjut ke baris pertama pada Algoritma 2.b. Label untuk simpul v_3 diperoleh dengan cara $\lambda(v_3) = k - \lambda(e_3) - \lambda(e_2) = 4$ (Gambar 3.8 (c)). Oleh karena label ini lebih besar dari $\lambda(v_1)$ dan label ini juga merupakan satu-satunya label yang masih tersedia, maka terbentuklah suatu PTSA C_3 dengan $k = 10$, yaitu dengan $\lambda(v_1) = 2$, $\lambda(v_2) = 6$, $\lambda(v_3) = 4$, $\lambda(e_1) = 3$, $\lambda(e_2) = 1$, $\lambda(e_3) = 5$. Label-label inilah yang akan dicetak sebagai keluaran (*output*).

Proses pada algoritma ini belum berhenti walaupun sudah dihasilkan suatu PTSA C_3 dengan nilai $k = 10$. Yang dilakukan selanjutnya adalah keluar dari fungsi $extendCycle(3)$ dan melakukan *backtracking* ke fungsi $extendCycle(2)$, yaitu menuju baris ke-13 dan ke-14 pada Algoritma 2.b. Kemudian, proses berlanjut menuju baris ke-6 untuk mencoba label yang belum digunakan sebagai label simpul v_2 . Jika sudah tidak ada lagi label yang mungkin untuk melabel simpul v_2 , maka kembali dilakukan *backtracking* ke fungsi $initializeCycle$, yaitu menuju baris ke-11 dan 12 pada Algoritma 2.a. Langkah yang dilakukan selanjutnya adalah menuju baris ke-4 untuk mencoba label yang belum digunakan sebagai label busur e_1 . Jika sudah tidak ada lagi label yang mungkin untuk melabel busur e_1 , maka kembali dilakukan *backtracking*, yaitu menuju baris ke-13 pada Algoritma 2.a. Setelah itu, proses berlanjut ke baris 1 untuk melabel simpul v_1 dengan label yang belum digunakan sebagai label simpul v_1 . Dengan cara yang sama seperti yang telah dijelaskan, lakukan langkah-langkah berikutnya sampai iterasi tidak mungkin dilanjutkan lagi. Ini berarti, telah diperoleh semua PTSA yang tidak isomorfik untuk graf lingkaran C_3 dengan nilai $k = 10$.

3.2 Algoritma PTSA untuk Graf Matahari

Algoritma PTSA untuk graf matahari $C_n \odot \bar{K}_1$ merupakan algoritma yang diperoleh dengan memodifikasi algoritma PTSA untuk graf lingkaran C_n yang telah diberikan pada Subbab sebelumnya. Seperti pada algoritma PTSA untuk graf C_n , yang menjadi masukan (*input*) dalam algoritma PTSA untuk graf $C_n \odot \bar{K}_1$

adalah n dan k , yang menyatakan ukuran dari graf matahari dan konstanta ajaib yang mungkin, yang bergantung pada nilai n , seperti yang diberikan pada (3.4). Karena pada graf $C_n \odot \bar{K}_1$, nilai $|V| = |E| = 2n$, maka himpunan label yang tersedia untuk melabel setiap simpul dan busur pada graf $C_n \odot \bar{K}_1$ adalah $\{1, 2, \dots, 4n\}$. Label-label ini harus digunakan tepat satu kali.



Gambar 3.9 Graf $C_3 \odot \bar{K}_1$

Pada Bab 2 telah diberikan definisi dari graf matahari $C_n \odot \bar{K}_1$ beserta dengan cara penamaan pada graf tersebut. Graf matahari $C_n \odot \bar{K}_1$ merupakan graf yang diperoleh dari graf lingkaran C_n dengan penambahan satu simpul berderajat satu di setiap simpul C_n . Simpul-simpul pada graf C_n disebut sebagai simpul dalam, ditulis dengan v_1, v_2, \dots, v_n , dan simpul-simpul selain simpul pada graf C_n disebut sebagai simpul luar, ditulis dengan u_1, u_2, \dots, u_n . Sementara untuk busur-busurnya ditulis sebagai pasangan kedua simpul ujungnya. Namun, pada algoritma PTSA yang akan diberikan, busur-busur pada graf matahari tidak ditulis sebagai pasangan kedua simpul ujungnya. Busur-busur pada graf lingkaran C_n (busur dalam) ditulis sebagai e_1, e_2, \dots, e_n dan busur-busur selain busur pada graf C_n (busur luar) ditulis sebagai s_1, s_2, \dots, s_n . Sedangkan untuk simpul-simpulnya tetap ditulis sebagai v dan u . Pada Gambar 3.9 diberikan contoh penamaan untuk graf matahari $C_3 \odot \bar{K}_1$.

Secara umum, algoritma PTSA untuk graf matahari serupa dengan algoritma PTSA untuk graf lingkaran, yaitu terdiri dari dua fungsi, fungsi inisialisasi (*initializeSun*) dan fungsi perluasan (*extendSun*). Kedua fungsi ini terbagi berdasarkan adanya kesamaan proses melabel elemen pada graf matahari.

Fungsi inialisasi pada algoritma PTSA graf lingkaran merupakan fungsi yang memberikan label untuk 3 elemen, yaitu satu simpul dan dua busur yang hadir pada simpul tersebut. Sedangkan fungsi inialisasi pada graf matahari, selain memberikan label untuk 3 elemen tersebut (elemen-elemen yang terdapat pada graf lingkaran, yaitu satu simpul dalam dan dua busur dalam yang hadir pada simpul dalam tersebut), fungsi ini juga memberikan label untuk 2 elemen yang tidak terdapat pada graf lingkaran. Dua elemen ini adalah satu busur luar yang hadir pada simpul dalam terkait dan satu simpul luar yang dilalui oleh busur luar tersebut.

Pada fungsi inialisasi untuk PTSA graf matahari $C_n \odot \bar{K}_1$, pertama-tama diberikan label untuk simpul dalam v_1 , busur dalam e_1 , dan busur dalam e_n dengan label-label yang tersedia, sebut masing-masing dengan $\lambda(v_1)$, $\lambda(e_1)$, dan $\lambda(e_n)$. Kemudian tandai label-label tersebut agar tidak dapat digunakan lagi. Label untuk busur luar s_1 dapat ditentukan karena label ini merupakan *determined label*, yaitu $\lambda(s_1) = k - \lambda(v_1) - \lambda(e_1) - \lambda(e_n)$. Jika $\lambda(s_1)$ masih tersedia, maka tandai $\lambda(s_1)$ agar label ini tidak dapat digunakan lagi dan elemen selanjutnya yang akan dilabel adalah simpul luar u_1 . Label untuk simpul luar u_1 ini merupakan *determined label*, yaitu $\lambda(u_1) = k - \lambda(s_1)$. Jika $\lambda(u_1)$ tersedia, maka tandai label ini agar tidak dapat digunakan lagi dan akan dipanggil fungsi perluasan untuk melabel elemen-elemen berikutnya. Akan tetapi, label-label yang merupakan *determined label* ini mungkin tidak tersedia. Seperti halnya pada graf lingkaran, tidak tersedianya *determined label* $\lambda(x)$ untuk elemen pada graf matahari juga disebabkan oleh 3 kondisi. Kondisi yang menjadi berbeda adalah jika $\lambda(x) > 4n$, maka $\lambda(x)$ tidak tersedia pada himpunan label. Ini merupakan akibat dari $|V| + |E| = 4n$ pada graf matahari $C_n \odot \bar{K}_1$. Jika salah satu dari 3 kondisi ini terpenuhi ($\lambda(x) < 1$, $\lambda(x) > 4n$, atau $\lambda(x)$ telah digunakan untuk melabel elemen pada graf matahari), maka akan dilakukan *backtracking*. Konsep *backtracking* yang digunakan pada algoritma ini sama seperti konsep *backtracking* yang digunakan pada algoritma PTSA untuk graf lingkaran, yaitu menelusuri langkah-langkah yang telah dilakukan secara mundur untuk mengganti label dari elemen yang dilabel sebelumnya sampai diperoleh label elemen yang tepat. Setiap kali melakukan penggantian label suatu elemen pada

proses *backtracking*, ubah dahulu status label yang digunakan sebelumnya untuk melabel elemen tersebut sehingga menjadi tersedia kembali.

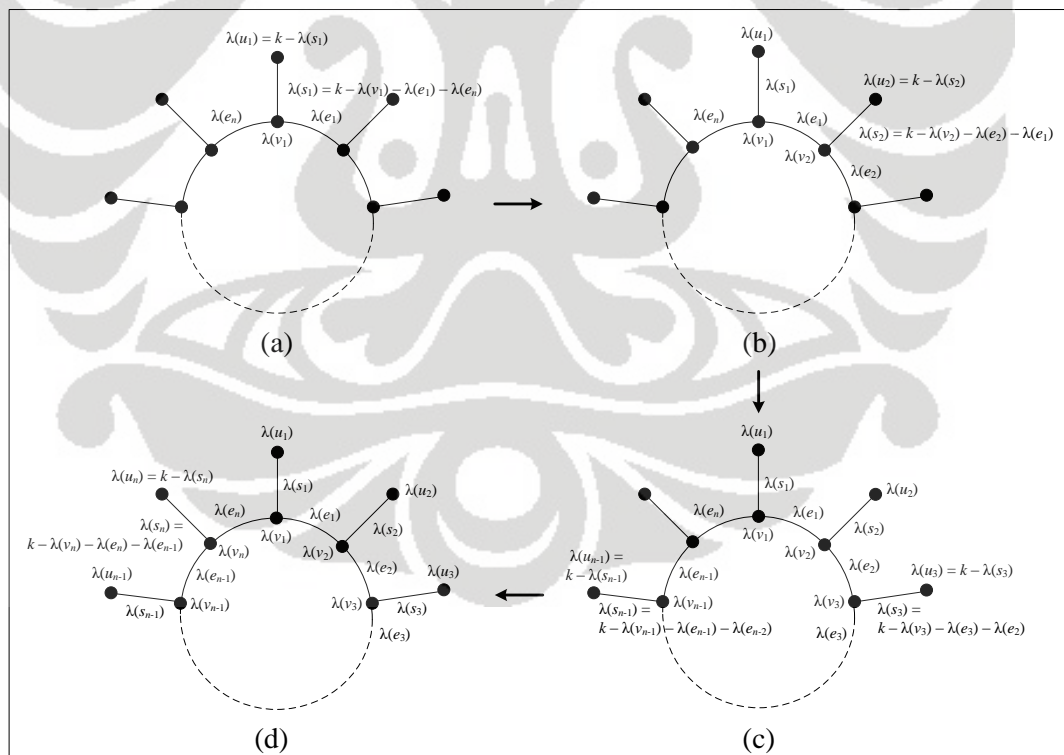
Fungsi perluasan pada graf lingkaran merupakan fungsi yang memberikan label untuk 2 elemen berikutnya, yaitu satu simpul dan satu busur. Simpul dan busur yang dilabel disini merupakan simpul dan busur yang berkaitan dengan salah satu busur yang telah dilabel sebelumnya. Fungsi perluasan ini akan memanggil dirinya sendiri dan pada tahap akhir pelabelan, akan terdapat satu simpul yang belum dilabel. Sementara, fungsi perluasan pada graf matahari, selain memberikan label untuk 2 elemen tersebut (elemen berikutnya pada graf lingkaran, yaitu satu simpul dalam dan satu busur dalam), fungsi ini juga memberikan label untuk satu busur luar dan satu simpul luar yang berkaitan. Pada tahap akhir pelabelan, selain terdapat satu simpul dalam yang belum dilabel, terdapat juga satu busur luar dan satu simpul luar yang belum dilabel.

Fungsi perluasan untuk graf matahari $C_n \odot \bar{K}_1$ menggunakan parameter t yang menyatakan posisi dari simpul dalam yang akan dilabel. Fungsi inisialisasi akan memanggil fungsi perluasan dengan parameter $t = 2$. Ini berarti, simpul dalam yang dilabel berikutnya adalah simpul v_2 , misalkan dilabel dengan $\lambda(v_2)$. Tandai $\lambda(v_2)$ agar label ini tidak dapat digunakan lagi. Elemen berikutnya yang dilabel adalah busur dalam e_2 , sebut dengan $\lambda(e_2)$, dan tandai label ini agar tidak dapat digunakan lagi. Label untuk busur luar s_2 merupakan *determined label*, yaitu $\lambda(s_2) = k - \lambda(v_2) - \lambda(e_2) - \lambda(e_1)$. Jika $\lambda(s_2)$ tidak tersedia, maka akan dilakukan *backtracking*. Jika $\lambda(s_2)$ tersedia, maka tandai label ini agar tidak dapat digunakan lagi dan elemen yang dilabel selanjutnya adalah simpul luar u_2 . Label untuk simpul luar u_2 ini merupakan *determined label*, yaitu $\lambda(u_2) = k - \lambda(s_2)$. Jika $\lambda(u_2)$ tersedia, maka tandai $\lambda(u_2)$ agar label ini tidak dapat digunakan lagi dan fungsi perluasan akan memanggil dirinya sendiri dengan parameter $t+1$. Jika $\lambda(u_2)$ tidak tersedia, maka akan dilakukan *backtracking*.

Proses rekursif pada fungsi perluasan akan berhenti ketika nilai $t = n$. Artinya, simpul dalam yang dilabel berikutnya adalah simpul v_n . Selain simpul dalam v_n , elemen-elemen yang dilabel pada proses ini adalah busur luar s_n dan simpul luar u_n . Label untuk busur luar s_n dan simpul luar u_n ini merupakan *determined label*. Label untuk busur luar s_n diperoleh dari $k - \lambda(v_n) - \lambda(e_n) - \lambda(e_{n-1})$.

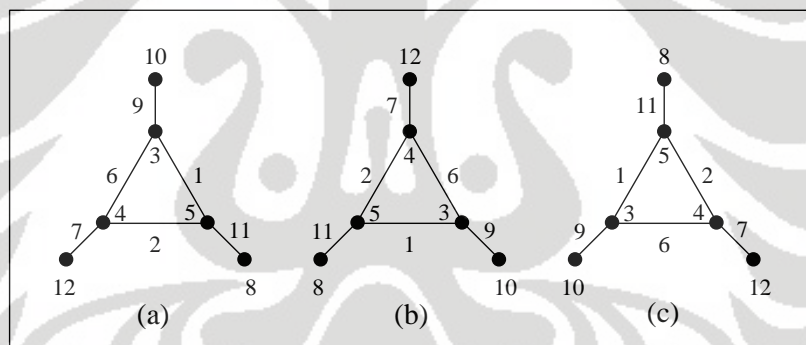
Sebut hasil ini dengan $\lambda(s_n)$. Jika $\lambda(s_n)$ tidak tersedia, maka akan dilakukan *backtracking*. Jika label $\lambda(s_2)$ tersedia, maka tandai label ini agar tidak dapat digunakan lagi dan beri label untuk simpul luar u_n , yaitu $\lambda(u_n) = k - \lambda(s_n)$. Jika $\lambda(u_n)$ tidak tersedia, maka dilakukan *backtracking*. Jika $\lambda(u_n)$ merupakan satu-satunya label yang masih tersedia, maka telah terbentuk suatu PTSA untuk graf matahari $C_n \odot \bar{K}_1$ dengan nilai konstanta ajaib k , sehingga label-label akan dicetak. Meskipun telah diperoleh satu PTSA, proses ini tetap berlanjut secara *backtracking* sampai dihasilkan semua PTSA yang mungkin untuk graf $C_n \odot \bar{K}_1$ dengan konstanta ajaib k .

Pada Gambar 3.10 diberikan ilustrasi pembentukan PTSA untuk graf lingkaran $C_n \odot \bar{K}_1$ dengan konstanta ajaib k . Gambar 3.10 (a) merupakan proses yang dilakukan pada fungsi inisialisasi, Gambar 3.10 (b), (c), dan (d) merupakan proses yang dilakukan pada fungsi perluasan. Pada ilustrasi ini, semua label yang merupakan *determined label* diasumsikan tersedia.



Gambar 3.10 Proses pembentukan PTSA pada graf $C_n \odot \bar{K}_1$

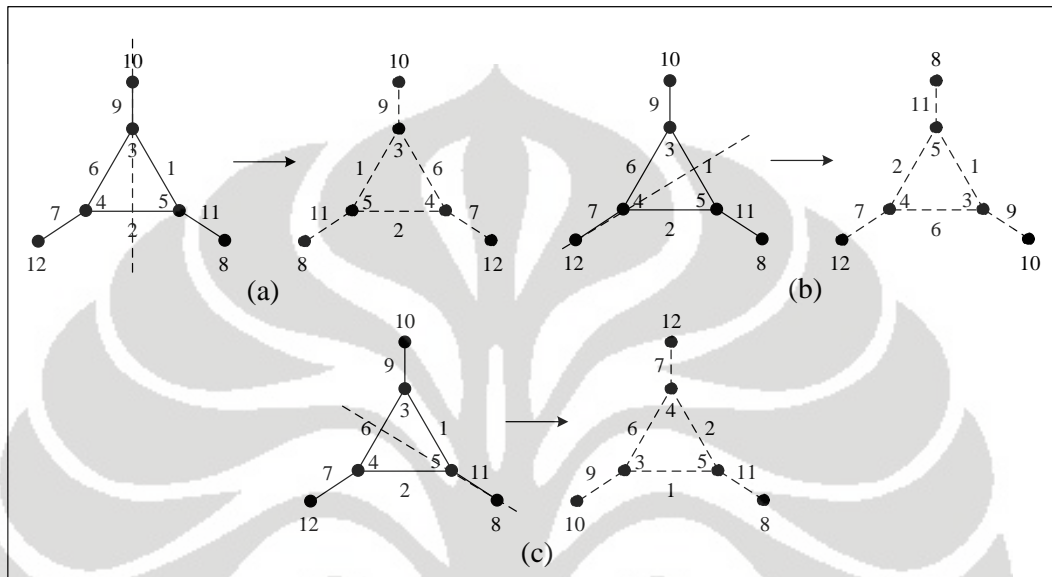
Agar proses yang dilakukan menjadi efektif, maka perlu dihindari kasus-kasus isomorfik yang terjadi pada PTSA graf matahari. Seperti pada pelabelan pada graf lingkaran, pelabelan yang isomorfik pada graf matahari juga dapat dihasilkan dengan melakukan rotasi atau refleksi terhadap suatu pelabelan yang ada. Pada Gambar 3.11 diberikan semua PTSA isomorfik untuk graf $C_3 \odot \bar{K}_1$ dengan $k = 19$ yang merupakan kasus rotasional simetris. Dari gambar-gambar tersebut dapat terlihat bahwa cara untuk menghindari PTSA isomorfik dengan kasus ini antara lain adalah salah satu label untuk simpul dalam harus lebih kecil dari label simpul-simpul dalam yang lain, atau salah satu label untuk simpul luar harus lebih kecil dari label simpul-simpul luar yang lain. Dalam hal ini, simpul dalam v_1 dipilih sebagai simpul dalam dengan label minimum. Syarat $\lambda(v_1)$ minimum ini diberikan pada fungsi perluasan karena fungsi perluasan menentukan label untuk simpul-simpul dalam lain selain simpul dalam v_1 . Dengan menggunakan syarat ini, PTSA pada Gambar 3.11 (b) dan (c) tidak mungkin terjadi.



Gambar 3. 11 PTSA $C_3 \odot \bar{K}_1$ yang isomorfik dengan kasus rotasional simetris

Pada Gambar 3.12 diberikan semua PTSA isomorfik untuk graf $C_3 \odot \bar{K}_1$ dengan $k = 19$ yang merupakan kasus refleksional simetris dengan sumbu simetri yang dinyatakan oleh garis putus-putus. Dari gambar ini dapat terlihat bahwa salah satu cara untuk menghindari PTSA hasil refleksi pada Gambar 3.12 (a) adalah dengan memberikan syarat bahwa pada dua busur dalam berlabel yang bertetangga, salah satu label busurnya lebih kecil dari label busur yang satu. Dalam hal ini, dipilih label untuk busur e_1 lebih kecil dari label untuk busur e_n ($\lambda(e_1) < \lambda(e_n)$). Syarat ini diberikan pada fungsi inisialisasi karena fungsi inisialisasi menentukan

label untuk e_1 dan e_n . Karena pada fungsi inialisasi diberikan syarat bahwa $\lambda(e_1) < \lambda(e_n)$ dan pada fungsi perluasan diberikan syarat bahwa label simpul dalam v_1 lebih kecil dari label simpul-simpul dalam lainnya, maka PTSA hasil refleksi pada Gambar 3.12 (b) dan (c) tidak mungkin terjadi.



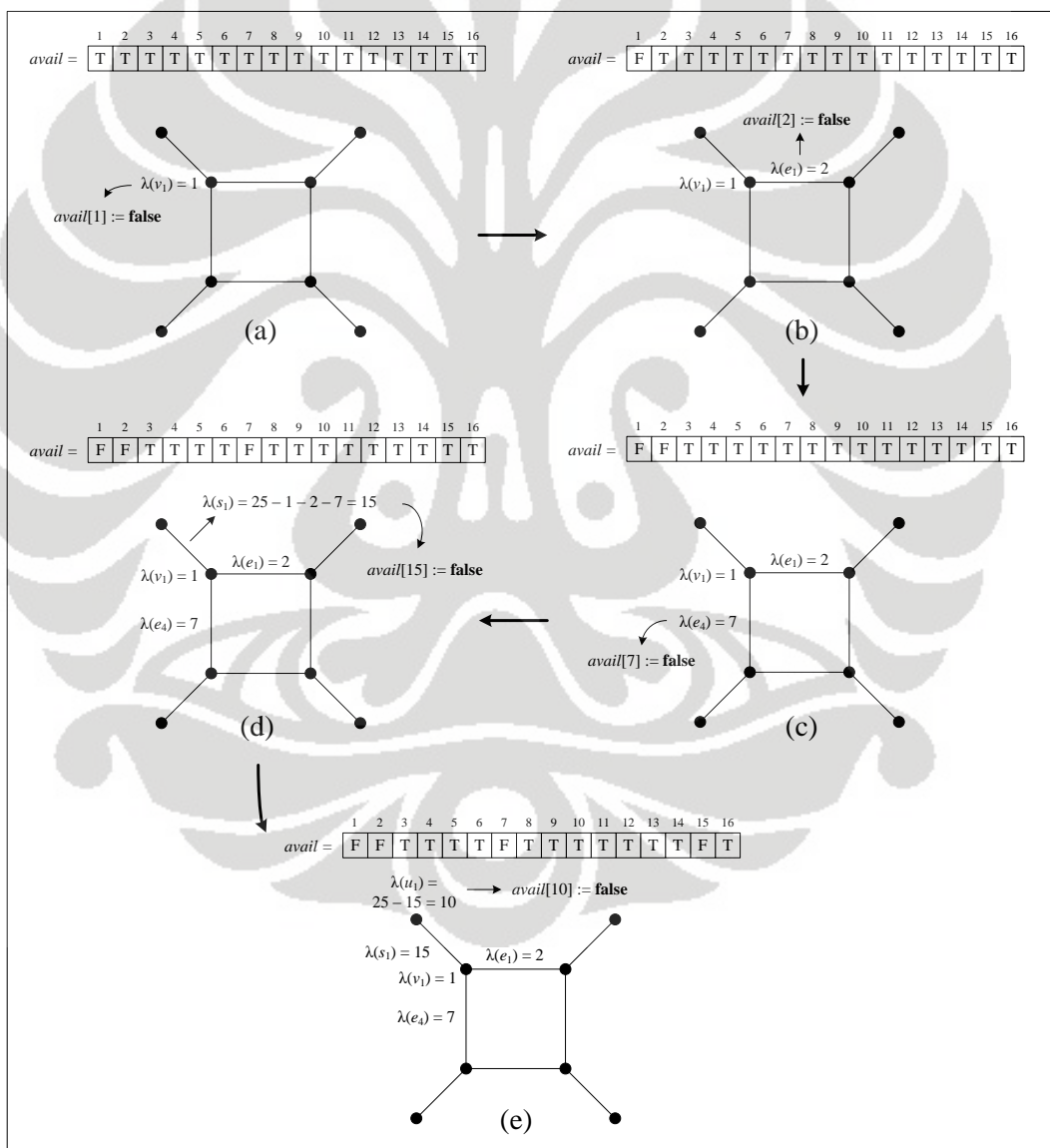
Gambar 3.12 PTSA $C_3 \odot \bar{K}_1$ yang isomorfik dengan kasus refleksional simetris

Oleh karena simpul dalam v_1 merupakan simpul dalam dengan label minimum, maka iterasi yang dilakukan pada fungsi inialisasi untuk memperoleh label simpul dalam ini dapat dikurangi dengan menambahkan syarat bahwa label maksimum untuk simpul dalam v_1 adalah $3n+1$. Apabila simpul v_1 merupakan simpul dalam yang dilabel dengan label $3n+1$, maka simpul-simpul dalam lainnya dapat dilabel dengan $n-1$ label tersisa pada himpunan label yang tersedia, yang lebih besar dari $3n+1$. Algoritma-algoritma PTSA pada graf matahari ini diberikan pada Algoritma 3, dimana Algoritma 3.a. merupakan fungsi inialisasi dan Algoritma 3.b. merupakan fungsi perluasan. *Avail* yang terdapat pada algoritma ini merupakan suatu *array* dengan ukuran $4n$. Seperti algoritma PTSA pada graf lingkaran, dapat diperhatikan bahwa pada algoritma PTSA pada graf matahari juga terjadi pemanggilan fungsi secara rekursif.

<p>Algoritma 3.a. Fungsi initializeSun</p> <pre> 1 for each available label i where $i \leq 3n+1$ do 2 $\lambda(v_1) := i$ 3 $avail[i] := false$ 4 for each available label p do 5 $\lambda(e_1) := p$ 6 $avail[p] := false$ 7 for each available label j where $j > \lambda(e_1)$ do 8 $\lambda(e_n) := j$ 9 $avail[j] := false$ 10 $\lambda(s_1) := k - \lambda(v_1) - \lambda(e_1) - \lambda(e_n)$ 11 if $0 < \lambda(s_1) \leq 4n$ and $avail[\lambda(s_1)]$ then 12 $avail[\lambda(s_1)] := false$ 13 $\lambda(u_1) := k - \lambda(s_1)$ 14 if $0 < \lambda(u_1) \leq 4n$ and $avail[\lambda(u_1)]$ then 15 $avail[\lambda(u_1)] := false$ 16 extendSun(2) 17 $avail[\lambda(u_1)] := true$ 18 end if 19 $avail[\lambda(s_1)] := true$ 20 end if 21 $avail[j] := true$ 22 end for 23 $avail[p] := true$ 24 end for 25 $avail[i] := true$ 26 end for </pre>	<p>Algoritma 3.b. Fungsi extendSun</p> <pre> 1 if $t = n$ then 2 for each available label i where $i > \lambda(v_1)$ do 3 $\lambda(v_n) := i$ 4 $avail[i] := false$ 5 $\lambda(s_n) := k - \lambda(v_n) - \lambda(e_n) - \lambda(e_{n-1})$ 6 if $0 < \lambda(s_n) \leq 4n$ and $avail[\lambda(s_n)]$ then 7 $avail[\lambda(s_n)] := false$ 8 $\lambda(u_n) := k - \lambda(s_n)$ 9 if $0 < \lambda(u_n) \leq 4n$ and $avail[\lambda(u_n)]$ then 10 Print() 11 end if 12 $avail[\lambda(s_n)] := true$ 13 end if 14 $avail[i] := true$ 15 end for 16 else 17 for each available label i where $i > \lambda(v_1)$ do 18 $\lambda(v_t) := i$ 19 $avail[i] := false$ 20 for each available label p do 21 $\lambda(e_t) := p$ 22 $avail[p] := false$ 23 $\lambda(s_t) := k - \lambda(v_t) - \lambda(e_t) - \lambda(e_{t-1})$ 24 if $0 < \lambda(s_t) \leq 4n$ and $avail[\lambda(s_t)]$ then 25 $avail[\lambda(s_t)] := false$ 26 $\lambda(u_t) := k - \lambda(s_t)$ 27 if $0 < \lambda(u_t) \leq 4n$ and $avail[\lambda(u_t)]$ then 28 $avail[\lambda(u_t)] := false$ 29 extendSun($t+1$) 30 $avail[\lambda(u_t)] := true$ 31 end if 32 $avail[\lambda(s_t)] := true$ 33 end if 34 $avail[p] := true$ 35 end for 36 $avail[i] := true$ 37 end for 38 end if </pre>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

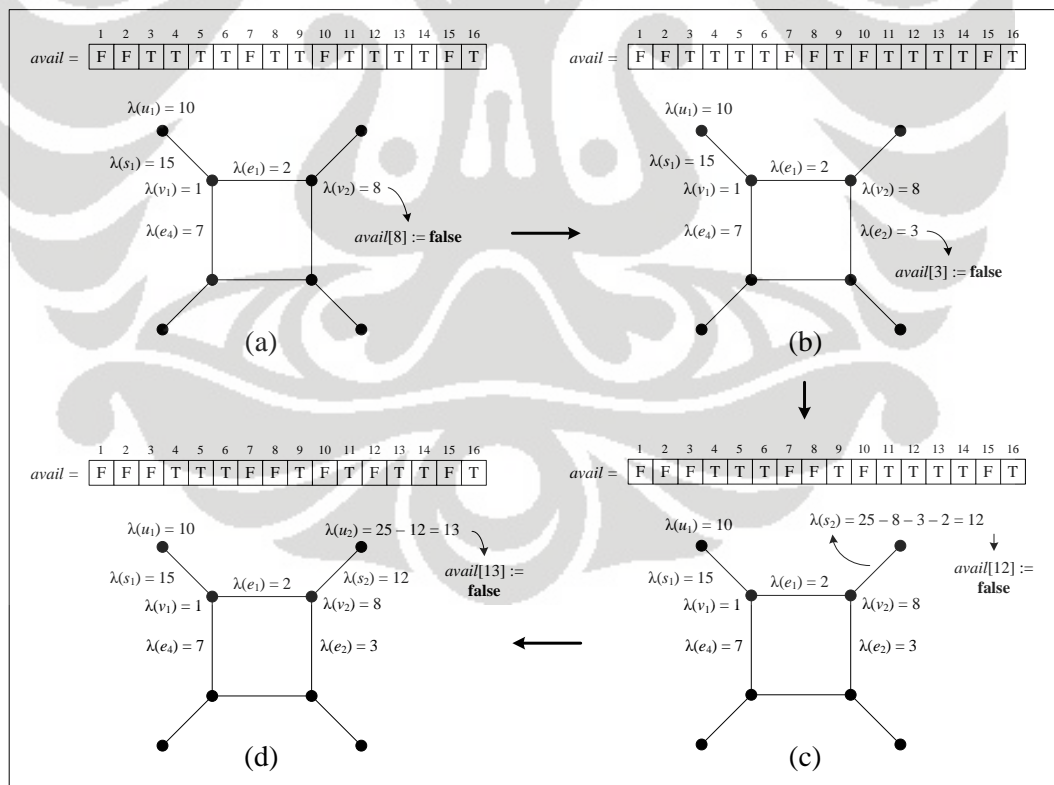
Sebagai contoh penggunaan dari algoritma PTSA ini akan dilakukan pelabelan untuk graf $C_4 \odot \bar{K}_1$ dengan nilai $k = 25$. Banyaknya label yang digunakan untuk membentuk PTSA S_4 adalah $4n = 4(4) = 16$, yaitu 1, 2, 3, ..., 16, sehingga $avail = [T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T]$. Menggunakan fungsi inisialisasi yang diberikan pada Algoritma 3.a., langkah pertama yang dilakukan adalah mencoba label yang tersedia untuk melabel simpul dalam v_1 . Misalkan pilih 1, maka $\lambda(v_1) = 1$ dan nyatakan $avail[1] = false$ (Gambar 3.13 (a)). Langkah selanjutnya adalah mencoba label yang tersedia untuk melabel busur dalam e_1 . Misalkan pilih 2, maka $\lambda(e_1) = 2$ dan nyatakan $avail[2] = false$ (Gambar 3.13 (b)). Elemen berikutnya yang dilabel adalah busur dalam e_4 . Label yang dipilih untuk melabel busur ini adalah yang lebih besar dari $\lambda(e_1) = 2$. Misalkan pilih 7, maka

$\lambda(e_4) = 7$ dan nyatakan $avail[7] = false$ (Gambar 3.13 (c)). Label untuk busur luar s_1 dapat ditentukan dengan menghitung $\lambda(s_1) = k - \lambda(v_1) - \lambda(e_1) - \lambda(e_4) = 15$. Karena label ini masih tersedia, maka nyatakan $avail[15] = false$ (Gambar 3.13 (d)). Label untuk simpul luar u_1 juga dapat ditentukan, yaitu $\lambda(u_1) = k - \lambda(s_1) = 10$. Karena label ini masih tersedia, maka nyatakan $avail[10] = false$ (Gambar 3.13 (e)). Kemudian, fungsi inialisasi ini akan memanggil fungsi *extendSun* dengan parameter $t = 2$. Sampai sejauh ini, label yang telah digunakan untuk melabeli adalah 1, 2, 7, 10, 15.

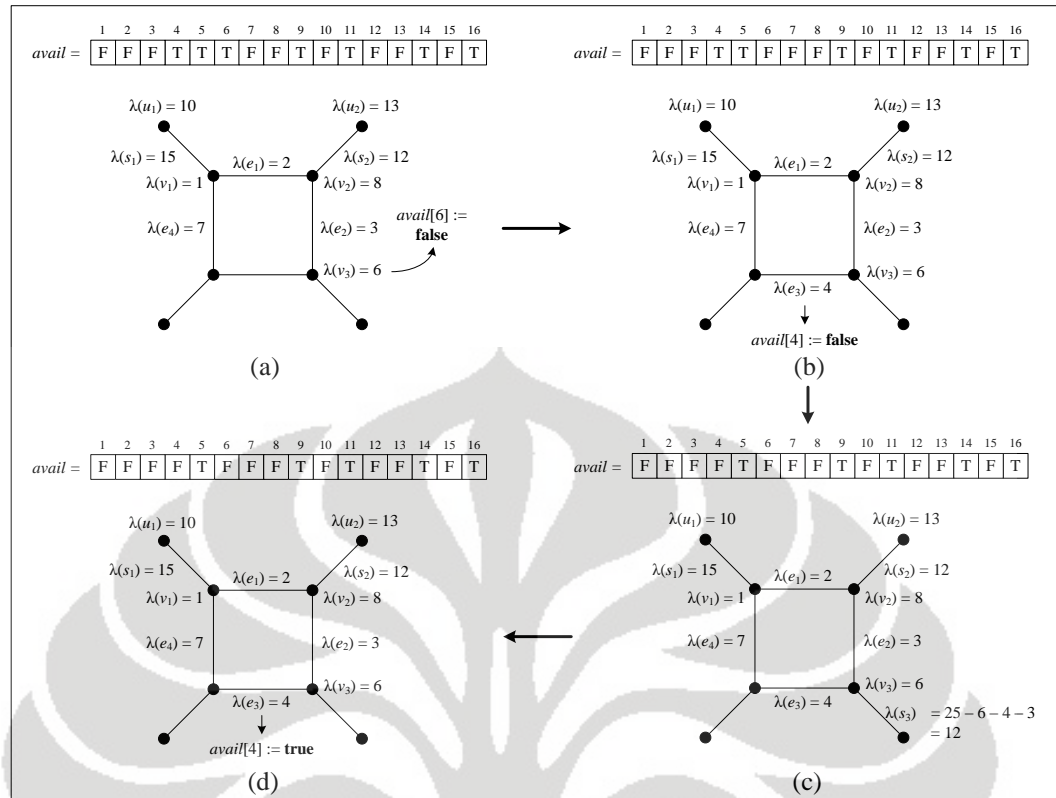


Gambar 3.13 Proses pada *initializeSun* untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$

Parameter 2 pada fungsi *extendSun*(2) menyatakan bahwa elemen graf $C_4 \odot \bar{K}_1$ yang dilabel berikutnya adalah simpul dalam ke-2. Karena simpul dalam ini bukan merupakan simpul dalam terakhir pada graf $C_4 \odot \bar{K}_1$, maka proses berlanjut ke baris 13 pada Algoritma 3.b. Label yang dipilih untuk melabel simpul dalam v_2 adalah yang lebih besar dari $\lambda(v_1) = 1$. Misalkan pilih 8, maka $\lambda(v_2) = 8$ dan nyatakan *avail*[8] = *false* (Gambar 3.14 (a)). Selanjutnya, misalkan pilih 3 sebagai label untuk busur dalam e_2 . Maka, $\lambda(e_2) = 3$ dan nyatakan *avail*[3] = *false* (Gambar 3.14 (b)). Label untuk busur luar s_2 dapat ditentukan, yaitu $\lambda(s_2) = k - \lambda(v_2) - \lambda(e_2) - \lambda(e_1) = 12$. Karena label ini masih tersedia, maka nyatakan *avail*[12] = *false* (Gambar 3.14 (c)). Label untuk simpul luar u_2 juga dapat ditentukan, yaitu $\lambda(u_2) = k - \lambda(s_2) = 13$. Karena label ini masih tersedia, maka nyatakan *avail*[13] = *false* (Gambar 3.14 (d)). Kemudian, fungsi *extendSun* ini akan memanggil dirinya sendiri dengan parameter $t+1 = 2+1 = 3$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 3, 7, 8, 10, 12, 13, 15.



Gambar 3.14 Proses pada *extendSun*(2) untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$

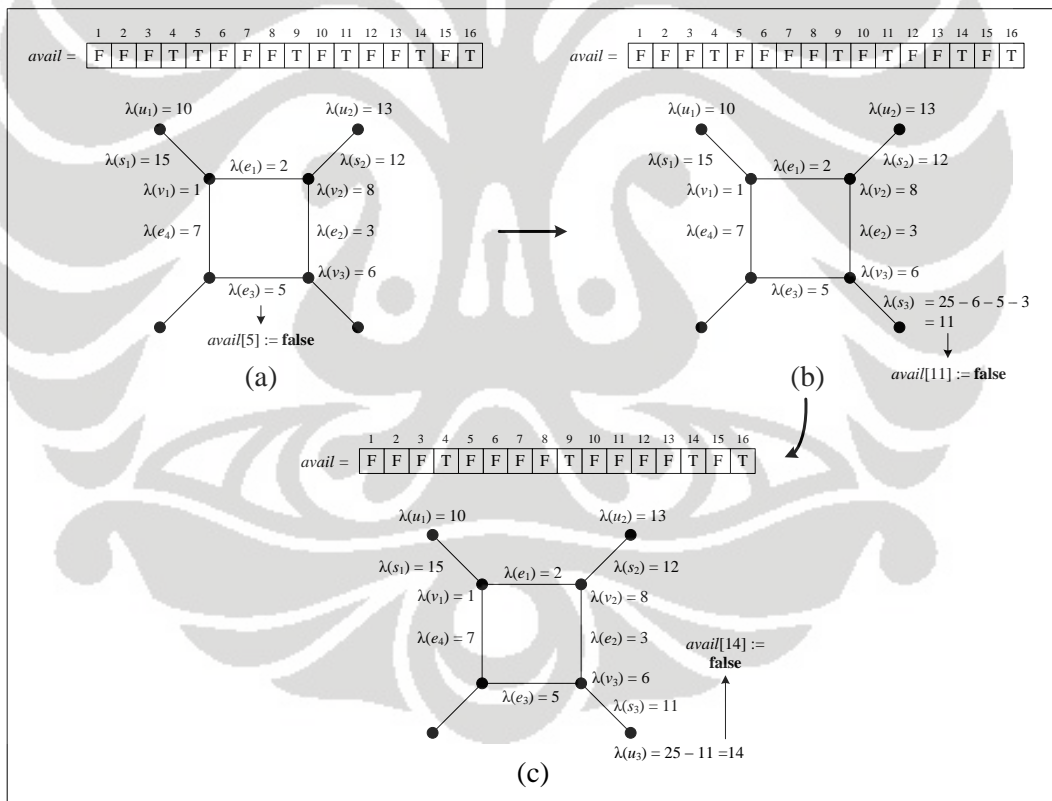


Gambar 3.15 Proses pada $extendSun(3)$ untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$

Parameter 3 pada fungsi $extendSun(3)$ menyatakan bahwa simpul dalam yang dilabel berikutnya adalah simpul dalam v_3 . Karena simpul dalam ini bukan merupakan simpul dalam terakhir pada graf $C_4 \odot \bar{K}_1$, maka proses berlanjut ke baris 13 pada Algoritma 3.b. Label yang dipilih untuk melabel simpul dalam v_3 adalah yang lebih besar dari $\lambda(v_1) = 1$. Misalkan pilih 6, maka $\lambda(v_3) = 6$ dan nyatakan $avail[6] = false$ (Gambar 3.15 (a)). Selanjutnya, elemen yang akan dilabel adalah busur dalam e_3 . Misalkan pilih 4, maka $\lambda(e_3) = 4$ dan nyatakan $avail[4] = false$ (Gambar 3.15 (b)). Label untuk busur luar s_3 dapat ditentukan, yaitu $\lambda(s_1) = k - \lambda(v_3) - \lambda(e_3) - \lambda(e_2) = 12$ (Gambar 3.15 (c)). Label 12 ternyata sudah tidak tersedia karena telah digunakan untuk melabel busur luar s_2 , maka proses menuju baris ke-29 pada Algoritma 3.b. untuk melakukan *backtracking*. Nyatakan $avail[4] = true$ yang berarti label 4 tersedia kembali. Label 4 ini merupakan label yang digunakan untuk melabel elemen yang dilabel sebelum melabel busur luar s_3 , yaitu busur dalam e_3 . Dengan kata lain, busur dalam e_3 gagal dilabel oleh 4

(Gambar 3.15 (d)). Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 3, 6, 7, 8, 10, 12, 13, 15.

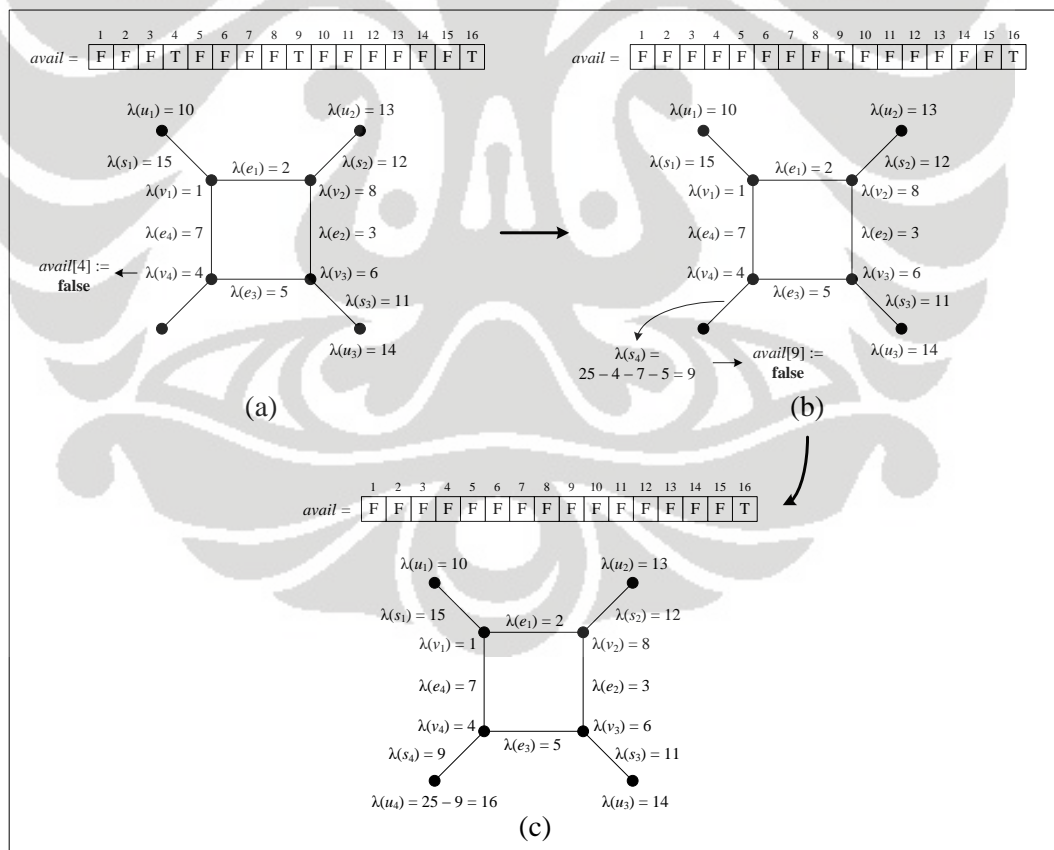
Proses selanjutnya adalah melabel kembali busur dalam e_3 dengan label lain yang masih tersedia (baris ke-17 pada Algoritma 3.b.). Misalkan pilih 5, maka $\lambda(e_3) = 5$ dan nyatakan $avail[5] = false$ (Gambar 3.16 (a)). Label untuk busur luar s_3 dapat ditentukan, yaitu $\lambda(s_3) = k - \lambda(v_3) - \lambda(e_3) - \lambda(e_2) = 11$. Karena label ini masih tersedia, maka nyatakan $avail[11] = false$ (Gambar 3.16 (b)). Label untuk simpul luar u_3 juga dapat ditentukan, yaitu $\lambda(u_3) = k - \lambda(s_3) = 14$. Karena label ini masih tersedia, maka nyatakan $avail[14] = false$ (Gambar 3.16 (c)). Fungsi *extendSun* ini kemudian akan memanggil dirinya sendiri dengan parameter $t+1 = 3+1 = 4$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15.



Gambar 3.16 Backtracking pada *extendSun(3)*

Parameter 4 pada fungsi *extendSun(4)* menyatakan bahwa simpul dalam yang akan dilabel adalah simpul dalam v_4 . Karena simpul dalam ini merupakan simpul dalam terakhir pada graf $C_4 \odot \bar{K}_1$, maka proses berlanjut ke baris pertama

pada Algoritma 3.b. Label yang dipilih untuk melabel simpul ini adalah yang lebih besar dari $\lambda(v_1) = 1$. Misalkan pilih 4, maka $\lambda(v_4) = 4$ dan nyatakan $avail[4] = false$ (Gambar 3.17 (a)). Kemudian, label untuk busur luar s_4 dapat ditentukan, yaitu $\lambda(s_4) = k - \lambda(v_4) - \lambda(e_4) - \lambda(e_3) = 9$. Karena label ini masih tersedia, maka nyatakan $avail[9] = false$ (Gambar 3.17 (b)). Label untuk simpul luar u_4 juga dapat ditentukan, yaitu $\lambda(u_4) = k - \lambda(s_4) = 16$ (Gambar 3.17 (c)). Karena label ini merupakan satu-satunya label yang masih tersedia, maka terbentuklah suatu PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$, yaitu dengan $\lambda(v_1) = 1, \lambda(v_2) = 8, \lambda(v_3) = 6, \lambda(v_4) = 4, \lambda(u_1) = 10, \lambda(u_2) = 13, \lambda(u_3) = 14, \lambda(u_4) = 16, \lambda(e_1) = 2, \lambda(e_2) = 3, \lambda(e_3) = 5, \lambda(e_4) = 7, \lambda(s_1) = 15, \lambda(s_2) = 12, \lambda(s_3) = 11, \lambda(s_4) = 9$. Label-label inilah yang dicetak sebagai keluaran. Walaupun telah terbentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$, proses pada algoritma tetap dilanjutkan secara *backtracking* sampai dihasilkan semua PTSA tidak isomorfik untuk graf $C_4 \odot \bar{K}_1$ dengan $k = 25$.

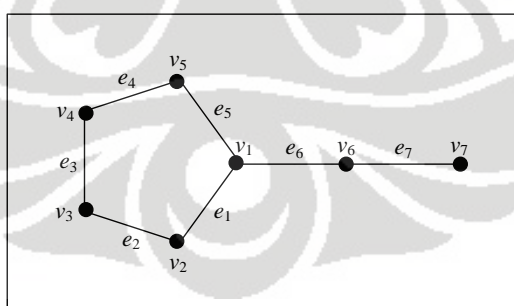


Gambar 3.17 Proses pada *extendSun*(4) untuk membentuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$

3.3 Algoritma PTSA untuk Graf Kecebong

Algoritma PTSA untuk graf kecebong $T_{m,n}$ merupakan algoritma yang diperoleh dengan memodifikasi algoritma PTSA untuk graf lingkaran C_n yang telah diberikan pada Subbab 3.1. Pada Bab 2 telah diberikan definisi dari graf kecebong beserta contohnya. Graf kecebong $T_{m,n}$ merupakan graf yang diperoleh dari graf lingkaran C_m dan graf lintasan P_n dengan penambahan satu busur yang menghubungkan salah satu simpul pada graf C_m dengan simpul awal atau simpul akhir pada graf P_n . Simpul-simpul pada graf kecebong ditulis dengan $v_1, v_2, \dots, v_m, v_{m+1}, \dots, v_{m+n}$ dan busur-busurnya ditulis sebagai pasangan kedua simpul ujungnya. Simpul v_1, v_2, \dots, v_m merupakan simpul-simpul pada graf lingkaran dan simpul $v_{m+1}, v_{m+2}, \dots, v_{m+n}$ merupakan simpul-simpul pada graf lintasan.

Pada algoritma PTSA yang akan diberikan, busur-busur pada graf kecebong tidak ditulis sebagai pasangan kedua simpul ujungnya, melainkan sebagai $e_1, e_2, \dots, e_m, e_{m+1}, \dots, e_{m+n}$. Busur e_1, e_2, \dots, e_m merupakan busur-busur yang terdapat pada graf C_m , busur e_{m+1} merupakan busur tambahan, dan busur $e_{m+2}, e_{m+3}, \dots, e_{m+n}$ merupakan busur-busur pada graf P_n . Sedangkan untuk simpul-simpulnya tetap ditulis sebagai $v_1, v_2, \dots, v_m, v_{m+1}, \dots, v_{m+n}$. Simpul v_1 dan v_{m+1} merupakan simpul-simpul yang dihubungkan dengan busur e_{m+1} . Gambar 3.18 memberikan contoh penamaan pada graf kecebong $T_{5,2}$.



Gambar 3.18 Graf $T_{5,2}$

Yang menjadi masukan (*input*) dalam algoritma PTSA untuk graf $T_{m,n}$ adalah m, n dan k , yang masing-masing menyatakan banyaknya simpul pada graf lingkaran, banyaknya simpul pada graf lintasan, dan konstanta ajaib yang mungkin, yang bergantung pada nilai m dan n , seperti yang diberikan pada (3.8). Kare-

na pada graf $T_{m,n}$, nilai $|V| = |E| = 2(m+n)$, maka himpunan label yang tersedia untuk melabel elemen-elemen pada graf $T_{m,n}$ adalah $\{1, 2, \dots, 2(m+n)\}$ dan setiap label ini harus digunakan tepat satu kali.

Dilihat dari struktur graf kecebong, selalu terdapat satu simpul dengan derajat tiga, yaitu simpul v_1 , dan satu simpul dengan derajat satu, yaitu simpul v_{m+n} . Sementara, simpul-simpul lainnya berderajat dua. Untuk menghitung bobot simpul v_1 , label yang berkontribusi adalah label simpul v_1 , busur e_1 , busur e_m , dan busur e_{m+1} . Untuk menghitung bobot simpul v_{m+n} , label yang berkontribusi adalah label simpul v_{m+n} dan busur e_{m+n} . Untuk menghitung bobot simpul-simpul lain yang berderajat dua, label yang berkontribusi adalah label simpul yang terkait dan label busur-busur yang hadir pada simpul tersebut.

Secara umum, algoritma PTSA untuk graf kecebong dimulai dengan mendapatkan label untuk simpul v_1 dan tiga label busur yang hadir pada simpul tersebut. Kemudian, elemen yang dilabel berikutnya adalah elemen-elemen yang terdapat pada graf lintasan. Setelah semua elemen pada graf lintasan ini diberi label, elemen yang dilabel berikutnya adalah elemen-elemen yang belum dilabel pada graf lingkaran. Berdasarkan proses melabel ini, algoritma PTSA untuk graf kecebong terdiri dari 3 fungsi, yaitu fungsi inisialisasi (*initializeTadpole*), fungsi perluasan (*extendTadpole*), dan fungsi finalisasi (*finalizeTadpole*). Fungsi perluasan dan fungsi finalisasi untuk PTSA graf kecebong menggunakan parameter t , dimana parameter ini pada masing-masing fungsi menyatakan posisi dari simpul pada graf lintasan yang akan dilabel dan posisi dari simpul pada graf lingkaran yang akan dilabel.

Pada fungsi inisialisasi untuk PTSA graf kecebong $T_{m,n}$, pertama-tama diberikan label untuk simpul v_1 , busur e_1 , dan busur e_m dengan label-label yang tersedia, misalkan masing-masing dilabel dengan $\lambda(v_1)$, $\lambda(e_1)$, dan $\lambda(e_m)$. Kemudian tandai label-label tersebut agar tidak dapat digunakan lagi. Label untuk busur $\lambda(e_{m+1})$ dapat ditentukan karena label ini merupakan *determined label*, yaitu $\lambda(e_{m+1}) = k - \lambda(v_1) - \lambda(e_1) - \lambda(e_m)$. Jika $\lambda(e_{m+1})$ masih tersedia, maka tandai $\lambda(e_{m+1})$ agar label ini tidak dapat digunakan lagi dan akan dipanggil fungsi perluasan dengan parameter $t = 1$, yang berarti elemen berikutnya yang akan dilabel adalah simpul pertama pada graf lintasan, yaitu simpul v_{m+1} . Jika $\lambda(e_{m+1})$ tidak

tersedia, maka akan dilakukan *backtracking*. Seperti halnya kondisi yang menyebabkan tidak tersedianya *determined label* untuk elemen pada graf-graf sebelumnya, tidak tersedianya *determined label* $\lambda(x)$ untuk elemen pada graf kecebong $T_{m,n}$ juga disebabkan oleh 3 kondisi. Kondisi yang menjadi berbeda adalah jika $\lambda(x) > 2(m+n)$, maka $\lambda(x)$ tidak tersedia pada himpunan label. Ini merupakan akibat dari $|V| + |E| = 2(m+n)$ pada graf $T_{m,n}$. Jika salah satu dari 3 kondisi ini terpenuhi ($\lambda(x) < 1$, $\lambda(x) > 2(m+n)$, atau $\lambda(x)$ telah digunakan untuk melabel elemen pada graf kecebong), maka akan dilakukan *backtracking*. Konsep *backtracking* yang digunakan pada algoritma ini juga sama seperti konsep *backtracking* yang sudah dibahas sebelumnya, yaitu menelusuri langkah-langkah yang telah dilakukan secara mundur untuk mengganti label dari elemen yang dilabel sebelumnya sampai diperoleh label elemen yang tepat. Setiap kali melakukan penggantian label suatu elemen pada proses *backtracking*, status label yang digunakan sebelumnya untuk melabel elemen tersebut harus diubah terlebih dahulu sehingga label tersebut menjadi tersedia kembali.

Fungsi perluasan pada graf kecebong $T_{m,n}$ merupakan fungsi yang akan memberikan label untuk elemen-elemen pada graf lintasan P_n . Proses melabel yang dilakukan pada fungsi perluasan ini serupa dengan proses melabel pada Algoritma 1.b., yaitu memberi label simpul dan selanjutnya menentukan label busur yang hadir pada simpul tersebut. Seperti yang telah disebutkan sebelumnya, fungsi perluasan untuk graf kecebong menggunakan parameter t yang menyatakan posisi simpul pada graf lintasan yang akan dilabel. Elemen yang pertama kali dilabel adalah simpul v_{m+t} , sebut dengan $\lambda(v_{m+t})$. Kemudian, tandai $\lambda(v_{m+t})$ agar label ini tidak dapat digunakan lagi. Label untuk busur e_{m+t+1} merupakan *determined label*, yaitu $\lambda(e_{m+t+1}) = k - \lambda(v_{m+t}) - \lambda(e_{m+t})$. Jika $\lambda(e_{m+t+1})$ tidak tersedia, maka akan dilakukan *backtracking*. Jika $\lambda(e_{m+t+1})$ masih tersedia, maka tandai $\lambda(e_{m+t+1})$ agar label ini tidak dapat digunakan lagi dan fungsi perluasan ini akan memanggil dirinya sendiri dengan parameter $t+1$. Ketika nilai $t = n$, akan terdapat satu simpul pada graf lintasan P_n yang belum dilabel, yaitu simpul v_{m+n} . Label untuk simpul ini merupakan *determined label*, yaitu $\lambda(v_{m+n}) = k - \lambda(e_{m+n})$. Jika $\lambda(v_{m+n})$ masih tersedia, maka fungsi perluasan akan memanggil fungsi finalisasi dengan parameter $t = 2$, yang artinya elemen yang akan dilabel berikutnya adalah simpul v_2

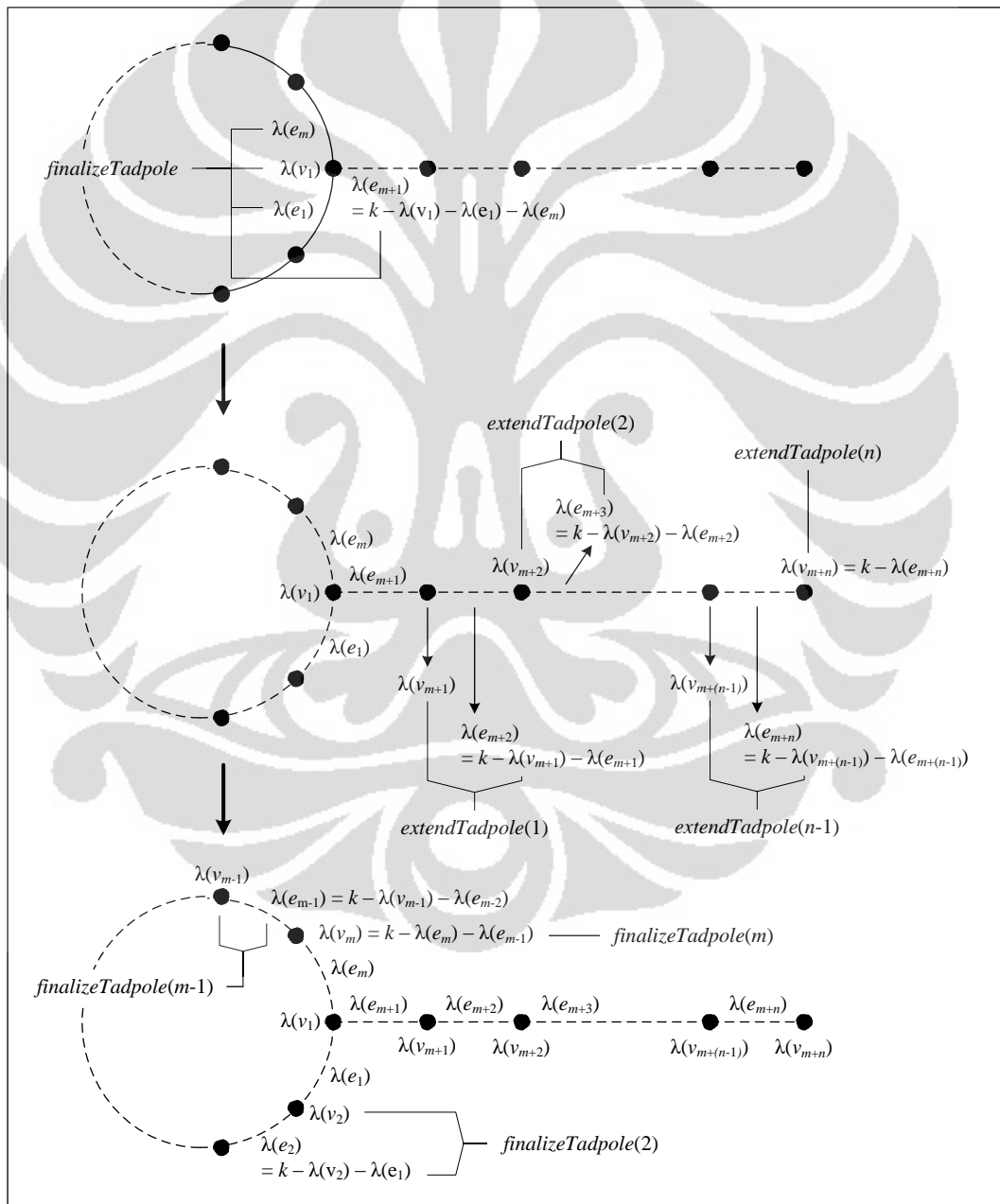
pada graf lingkaran. Jika $\lambda(e_{m+n})$ tidak tersedia, maka akan dilakukan *backtracking*.

Fungsi finalisasi pada graf kecebong $T_{m,n}$ merupakan fungsi yang akan memberikan label untuk elemen-elemen pada graf lingkaran C_m . Proses melabel yang dilakukan pada fungsi finalisasi ini sama dengan proses melabel pada Algoritma 1.b., yaitu memberikan label untuk simpul dan selanjutnya menentukan label untuk busur yang hadir pada simpul tersebut. Seperti yang telah disebutkan sebelumnya, pada fungsi ini digunakan parameter t sebagai posisi dari simpul pada graf lingkaran yang akan dilabel. Elemen yang pertama kali dilabel adalah simpul v_t , misalkan dilabel dengan $\lambda(v_t)$. Kemudian, tandai $\lambda(v_t)$ agar label ini tidak dapat digunakan lagi. Label untuk busur e_t dapat ditentukan karena label busur ini merupakan *determined label*, yaitu $\lambda(e_t) = k - \lambda(v_t) - \lambda(e_{t-1})$. Jika $\lambda(e_t)$ tidak tersedia, maka akan dilakukan *backtracking*. Jika $\lambda(e_t)$ tersedia, maka fungsi finalisasi ini akan memanggil dirinya sendiri dengan parameter $t+1$. Pada saat nilai $t = m$, akan terdapat satu simpul yang belum dilabel, yaitu simpul v_m . Label untuk simpul ini merupakan *determined label*, yaitu $\lambda(v_m) = k - \lambda(e_m) - \lambda(e_{m-1})$. Jika $\lambda(v_m)$ tidak tersedia, maka akan dilakukan *backtracking*. Jika $\lambda(v_m)$ merupakan satu-satunya label yang masih tersedia, maka telah terbentuk suatu PTSA untuk graf kecebong $T_{m,n}$ dengan nilai konstanta ajaib k , sehingga label-label akan dicetak. Meskipun telah dihasilkan suatu PTSA, proses akan tetap berlanjut secara *backtracking* sampai dihasilkan semua PTSA yang mungkin untuk graf $T_{m,n}$ dengan konstanta ajaib k .

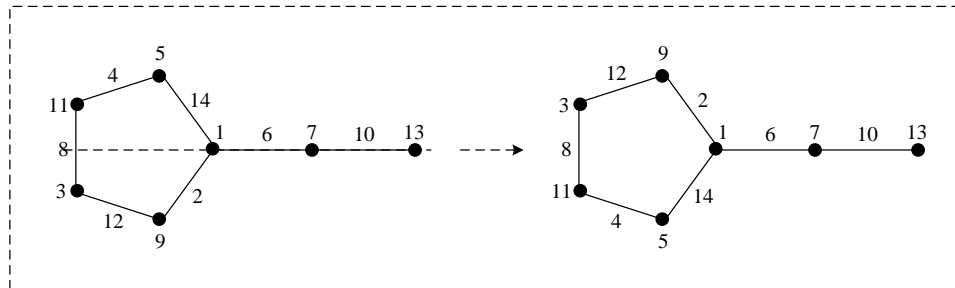
Pada Gambar 3.19 diberikan ilustrasi pembentukan PTSA untuk graf kecebong $T_{m,n}$ dengan konstanta ajaib k . Pada ilustrasi tersebut terdapat tiga gambar yang masing-masing menyatakan proses melabel elemen pada graf $T_{m,n}$ yang dilakukan pada fungsi inialisasi (*initializeTadpole*), fungsi perluasan (*extendTadpole(t)*), dan fungsi finalisasi (*finalizeTadpole(t)*). Semua label yang merupakan *determined label* diasumsikan tersedia.

Agar proses yang dilakukan menjadi efektif, maka perlu dihindari kasus isomorfik yang terjadi pada PTSA graf kecebong. Pada awal Bab ini telah disebutkan bahwa kasus isomorfik yang mungkin terjadi pada PTSA graf kecebong adalah kasus refleksional simetris. Pada Gambar 3.20 diberikan 2 PTSA isomor-

fik untuk graf $T_{5,2}$ dengan $k = 23$ dengan sumbu simetri yang dinyatakan oleh garis putus-putus. Dari kedua gambar tersebut dapat terlihat bahwa salah satu cara untuk menghindari PTSA isomorfik ini adalah dengan memberikan syarat bahwa label untuk busur e_1 lebih kecil dari label untuk busur e_m ($\lambda(e_1) < \lambda(e_m)$). Syarat ini diberikan pada fungsi inisialisasi karena fungsi inisialisasi menentukan label untuk e_1 dan e_m . Dengan menggunakan syarat ini, PTSA hasil refleksi pada Gambar 3.20 tidak mungkin terjadi.



Gambar 3. 19 Proses pembentukan PTSA pada graf $T_{m,n}$



Gambar 3. 20 PTSA isomorfik untuk graf $T_{5,2}$

Algoritma 4.a. Fungsi initializeTadpole

```

1 for each available label  $i$  do
2    $\lambda(v_1) := i$ 
3    $avail[i] := false$ 
4   for each available label  $j$  do
5      $\lambda(e_1) := j$ 
6      $avail[j] := false$ 
7     for each available label  $p$  where  $p > \lambda(e_1)$  do
8        $\lambda(e_m) := p$ 
9        $avail[p] := false$ 
10       $\lambda(e_{m+1}) := k - \lambda(v_1) - \lambda(e_1) - \lambda(e_m)$ 
11      if  $0 < \lambda(e_{m+1}) \leq 2(m+n)$  and  $avail[\lambda(e_{m+1})]$  then
12         $avail[\lambda(e_{m+1})] := false$ 
13        extendTadpole(1)
14         $avail[\lambda(e_{m+1})] := true$ 
15      end if
16    end for
17     $avail[p] := true$ 
18  end for
19   $avail[j] := true$ 
20 end for
21  $avail[i] := false$ 
22 end for

```

Algoritma 4.b. Fungsi extendTadpole(t)

```

1 if  $t = n$  then
2    $\lambda(v_{m+n}) := k - \lambda(e_{m+n})$ 
3   if  $0 < \lambda(v_{m+n}) \leq 2(m+n)$  and  $avail[\lambda(v_{m+n})]$  then
4      $avail[\lambda(v_{m+n})] := false$ 
5     finalizeTadpole(2)
6      $avail[\lambda(v_{m+n})] := true$ 
7   end if
8 else
9   for each available label  $i$  then
10     $\lambda(v_{m+i}) := i$ 
11     $avail[i] := false$ 
12     $\lambda(e_{m+i+1}) := k - \lambda(v_{m+i}) - \lambda(e_{m+i})$ 
13    if  $0 < \lambda(e_{m+i+1}) \leq 2(m+n)$  and  $avail[\lambda(e_{m+i+1})]$  then
14       $avail[\lambda(e_{m+i+1})] := false$ 
15      extendTadpole( $t+1$ )
16       $avail[\lambda(e_{m+i+1})] := true$ 
17    end if
18  end for
19   $avail[i] := true$ 
20 end if

```

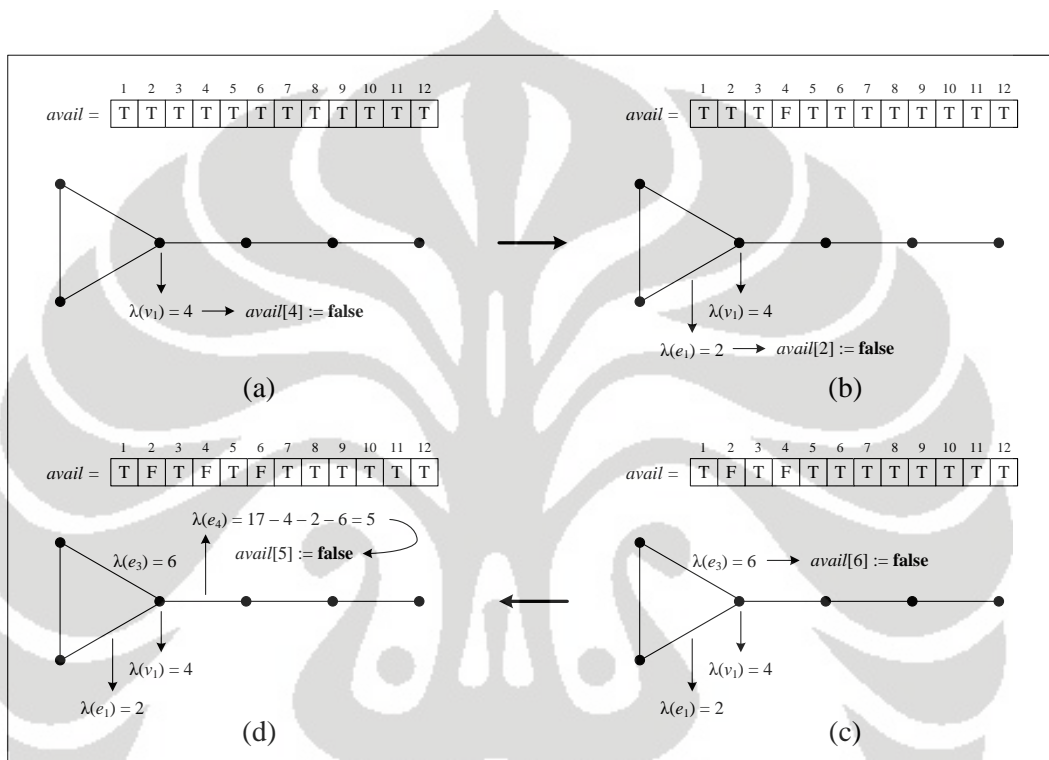
Algoritma 4.c. Fungsi finalizeTadpole(t)

```

1 if  $t = m$  then
2    $\lambda(v_m) := k - \lambda(e_m) - \lambda(e_{m-1})$ 
3   if  $0 < \lambda(v_m) \leq 2(m+n)$  and  $avail[\lambda(v_m)]$  then
4     Print()
5   end if
6 else
7   for each available label  $i$  then
8      $\lambda(v_t) := i$ 
9      $avail[i] := false$ 
10     $\lambda(e_t) := k - \lambda(v_t) - \lambda(e_{t-1})$ 
11    if  $0 < \lambda(e_t) \leq 2(m+n)$  and  $avail[\lambda(e_t)]$  then
12       $avail[\lambda(e_t)] := false$ 
13      finalizeTadpole( $t+1$ )
14       $avail[\lambda(e_t)] := true$ 
15    end if
16  end for
17   $avail[i] := true$ 
18 end if

```

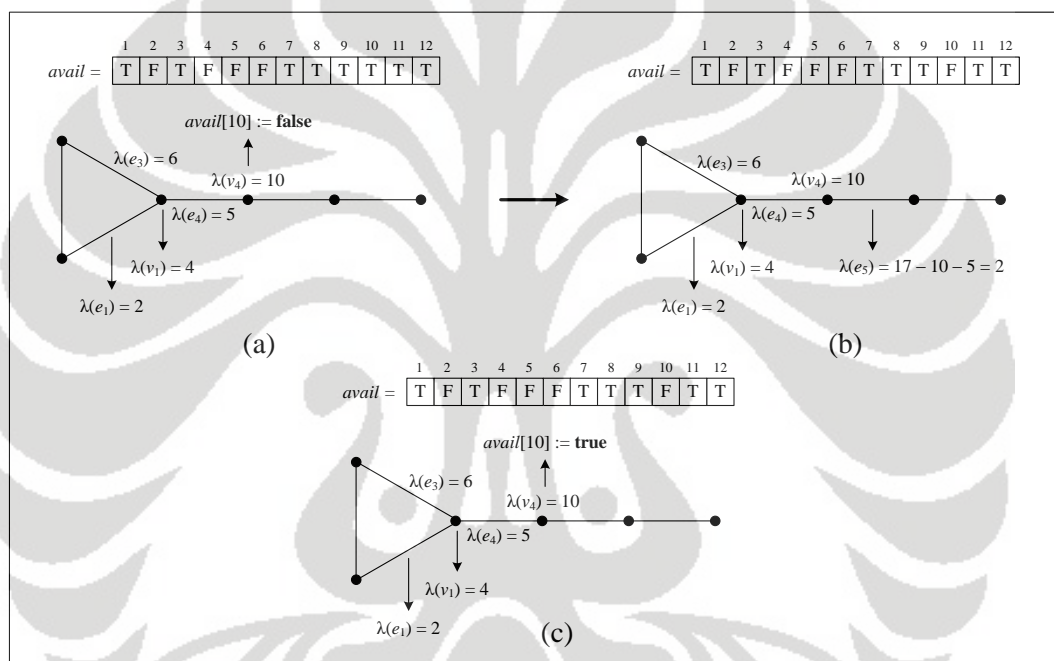
Algoritma-algoritma PTSA pada graf kecebong $T_{m,n}$ diberikan pada Algoritma 4, dimana Algoritma 4.a. merupakan fungsi inialisasi, Algoritma 4.b. merupakan fungsi perluasan, dan Algoritma 4.c. merupakan fungsi finalisasi. *Avail* yang terdapat pada algoritma ini merupakan suatu *array* dengan besar $2(m+n)$. Dari ketiga algoritma yang diberikan dapat diperhatikan bahwa pada algoritma-algoritma ini terjadi pemanggilan fungsi secara rekursif.



Gambar 3.21 Proses pada *initializeTadpole* untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$

Sebagai contoh penggunaan dari algoritma PTSA ini akan dilakukan pelabelan untuk graf $T_{3,3}$ dengan nilai $k = 17$. Artinya graf kecebong ini merupakan gabungan dari graf C_3 dan graf P_3 dengan tambahan satu busur. Banyaknya label yang digunakan untuk membentuk PTSA $T_{3,3}$ adalah $2(m+n) = 2(3+3) = 12$, yaitu $1, 2, 3, \dots, 12$, sehingga $avail = [T, T, T, T, T, T, T, T, T, T, T, T]$. Menggunakan fungsi inialisasi yang diberikan pada Algoritma 4.a., langkah pertama yang dilakukan adalah mencoba label yang tersedia untuk melabel simpul v_1 . Misalkan pilih 4, maka $\lambda(v_1) = 4$ dan nyatakan $avail[4] = false$ (Gambar 3.21 (a)). Kemudian, elemen selanjutnya yang dilabel adalah busur e_1 . Misalkan pilih 2, maka $\lambda(e_1) = 2$ dan nyatakan $avail[2] = false$ (Gambar 3.21 (b)). Langkah selanjutnya

adalah memberi label untuk busur e_3 . Label yang dipilih untuk melabel busur ini adalah yang lebih besar dari $\lambda(e_1) = 2$. Misalkan pilih 6, maka $\lambda(e_3) = 6$ dan nyatakan $avail[6] = false$ (Gambar 3.21 (c)). Label untuk busur e_4 dapat ditentukan dengan menghitung $k - \lambda(v_1) - \lambda(e_1) - \lambda(e_3)$, sehingga diperoleh $\lambda(e_4) = 5$. Karena label ini masih tersedia, maka nyatakan $avail[5] = false$ (Gambar 3.21 (d)). Kemudian, fungsi inisialisasi ini akan memanggil fungsi *extendTadpole* dengan parameter $t = 1$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 2, 4, 5, dan 6.

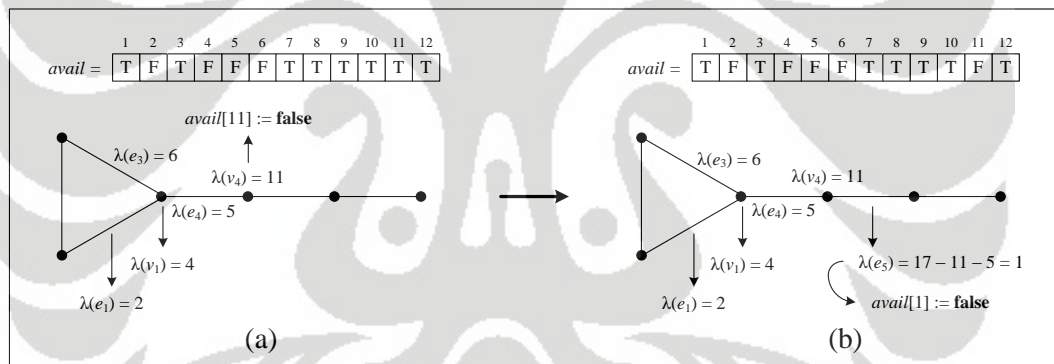


Gambar 3.22 Proses pada *extendTadpole*(1) untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$

Parameter 1 pada fungsi *extendTadpole*(1) menyatakan bahwa elemen graf $T_{3,3}$ yang dilabel berikutnya adalah simpul v_4 . Karena simpul v_4 ini merupakan simpul pertama pada graf lintasan, maka proses berlanjut ke baris 7 pada Algoritma 4.b. Misalkan pilih 10 untuk melabel simpul v_4 , maka $\lambda(v_4) = 10$ dan nyatakan $avail[10] = false$ (Gambar 3.22 (a)). Label untuk busur e_5 dapat ditentukan, yaitu $\lambda(e_5) = k - \lambda(v_4) - \lambda(e_4) = 2$ (Gambar 3.22 (b)). Label 2 ini ternyata sudah tidak tersedia karena telah digunakan sebagai label untuk busur e_1 , maka proses selanjutnya adalah menuju ke baris 16 pada Algoritma 4.b. untuk melakukan *back-*

tracking. Nyatakan $avail[10] = true$ yang berarti label 10 tersedia kembali. Label 10 ini merupakan label yang digunakan untuk melabel elemen yang dilabel sebelum melabel busur e_5 , yaitu simpul v_4 . Dengan kata lain, simpul v_4 gagal dilabel oleh 10 (Gambar 3.22 (c)). Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 2, 4, 5, dan 6.

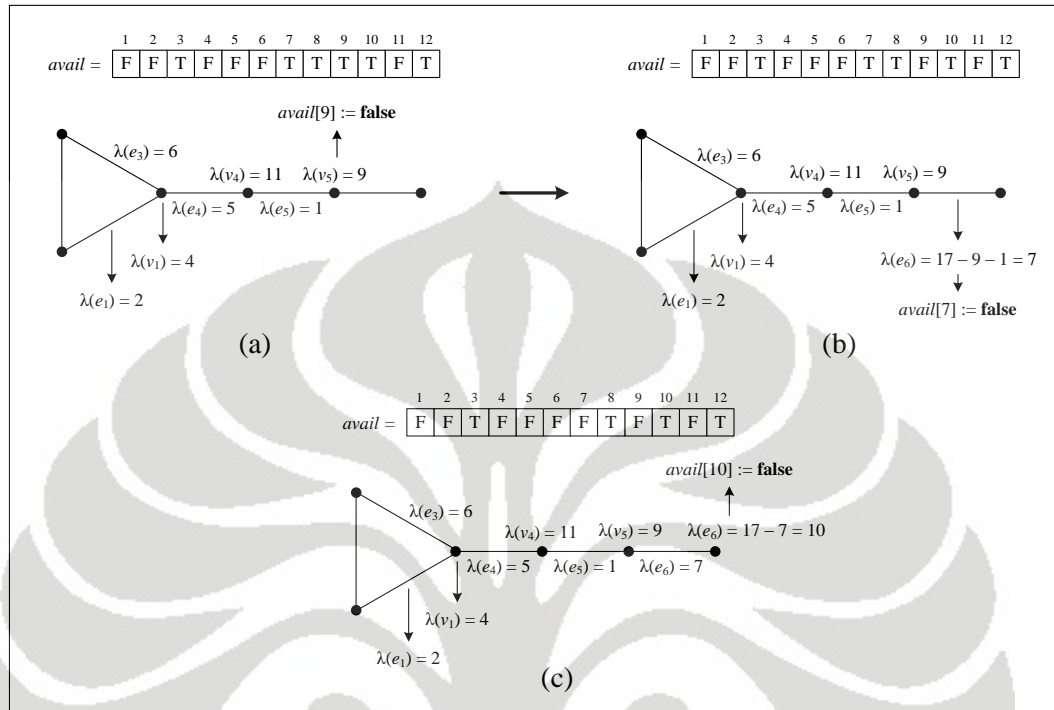
Proses selanjutnya adalah menuju baris ke- 8 pada Algoritma 4.b., yaitu simpul v_4 akan dilabel kembali dengan label lain yang masih tersedia. Misalkan pilih 11, maka $\lambda(v_4) = 11$ dan nyatakan $avail[11] = false$ (Gambar 3.23 (a)). Label untuk busur e_5 dapat ditentukan, yaitu $\lambda(e_5) = k - \lambda(v_4) - \lambda(e_4) = 1$. Karena label ini masih tersedia, maka nyatakan $avail[1] = false$ (Gambar 3.23 (b)). Fungsi *extendTadpole* ini kemudian akan memanggil dirinya sendiri dengan parameter $t+1 = 1+1 = 2$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 4, 5, 6, dan 11.



Gambar 3. 23 Backtracking pada *extendTadpole*(1)

Parameter 2 pada fungsi *extendTadpole*(2) menyatakan bahwa simpul pada graf lintasan yang dilabel berikutnya adalah simpul v_5 . Simpul v_5 ini merupakan simpul ke-2 pada graf lintasan. Karena simpul ini bukan simpul terakhir pada graf P_3 , maka proses berlanjut ke baris 7 pada Algoritma 4.b. Misalkan pilih 9 untuk melabel simpul v_5 , maka $\lambda(v_5) = 9$ dan nyatakan $avail[9] = false$ (Gambar 3.24 (a)). Label untuk busur e_6 dapat ditentukan, yaitu $\lambda(e_6) = k - \lambda(v_5) - \lambda(e_5) = 7$. Karena label ini masih tersedia, maka nyatakan $avail[7] = false$ (Gambar 3.24 (b)). Selanjutnya, fungsi *extendTadpole* akan memanggil dirinya sendiri dengan

parameter $t+1 = 2+1 = 3$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 4, 5, 6, 7, 9, dan 11.

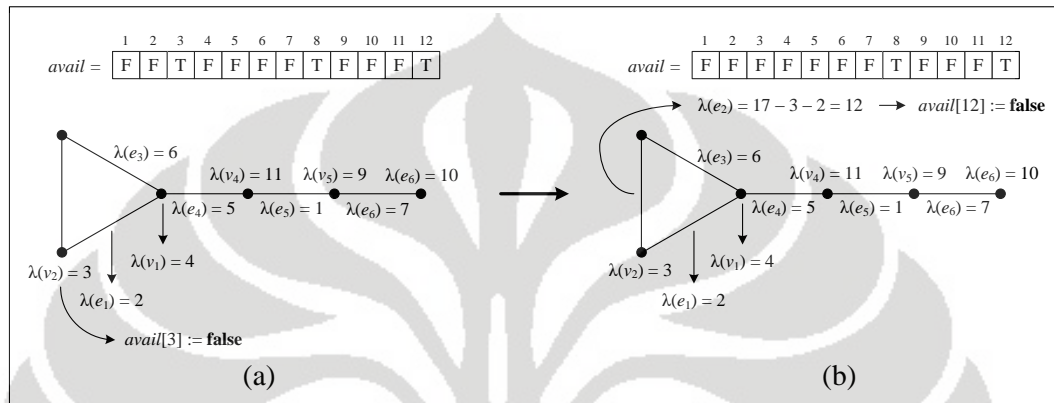


Gambar 3.24 Proses pada *extendTadpole(2)* dan *extendTadpole(3)* untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$

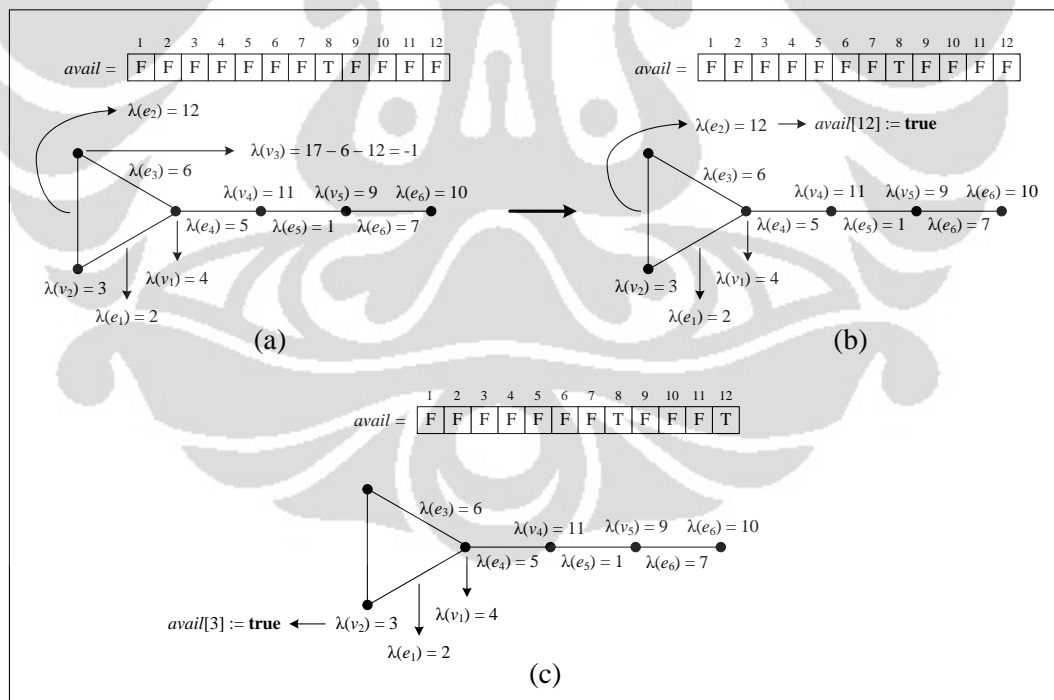
Parameter 3 pada fungsi *extendTadpole(3)* menyatakan bahwa simpul pada graf lintasan yang dilabel berikutnya adalah simpul v_6 . Karena simpul v_6 ini merupakan simpul terakhir pada graf lintasan P_3 , maka proses berlanjut ke baris pertama pada Algoritma 4.b. Label untuk simpul v_6 diperoleh dengan cara $k - \lambda(e_6)$, sehingga diperoleh $\lambda(v_6) = 10$. Karena label 10 ini masih tersedia, maka tandai $avail[10] = false$ (Gambar 3.24 (c)) dan fungsi *extendTadpole* akan memanggil fungsi *finalizeTadpole* dengan parameter $t = 2$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 4, 5, 6, 7, 9, 10, dan 11.

Parameter 2 pada fungsi *finalizeTadpole(2)* menyatakan bahwa simpul yang dilabel berikutnya adalah simpul pada graf lingkaran, yaitu simpul v_2 . Karena simpul ini bukan merupakan simpul terakhir pada graf C_3 , maka proses berlanjut ke baris 5 pada Algoritma 4.c. Misalkan pilih 3 untuk melabel simpul v_2 , maka $\lambda(v_2) = 3$ dan nyatakan $avail[3] = false$ (Gambar 3.25 (a)). Label untuk bu-

sur e_2 dapat ditentukan, yaitu $\lambda(e_2) = k - \lambda(v_2) - \lambda(e_1) = 12$. Karena label ini masih tersedia, maka nyatakan $avail[12] = false$ (Gambar 3.25 (b)). Selanjutnya, fungsi *finalizeTadpole* akan memanggil dirinya sendiri dengan parameter $t+1 = 2+1 = 3$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, dan 12.

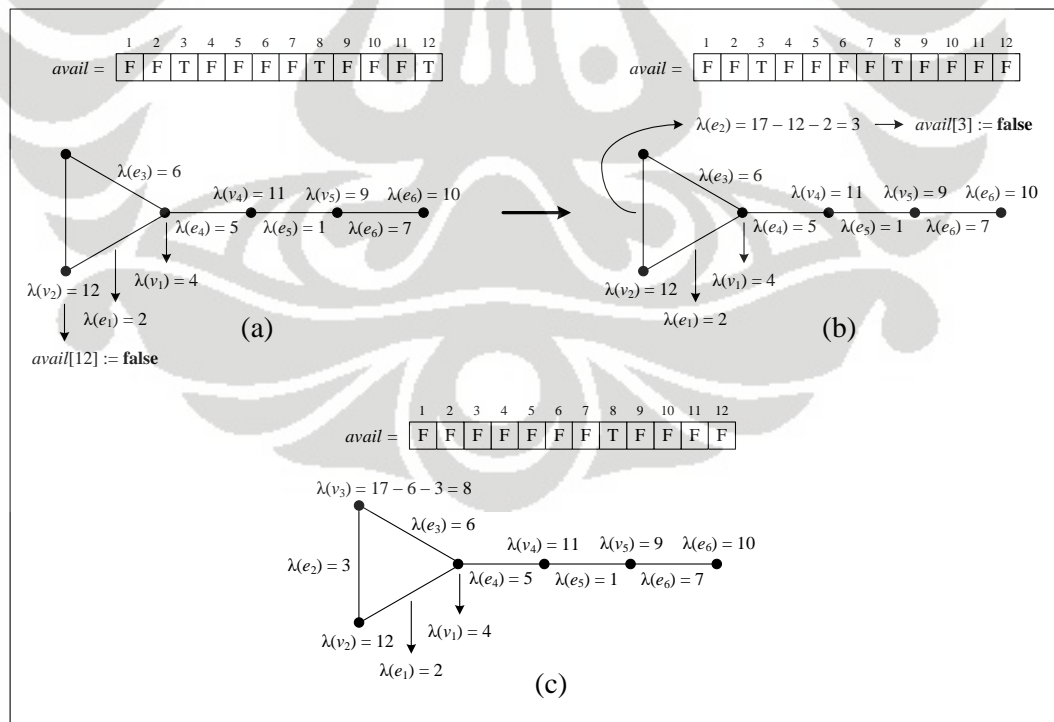


Gambar 3. 25 Proses pada finalizeTadpole(2) untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$



Gambar 3. 26 Backtracking pada finalizeTadpole(3)

Parameter 3 pada fungsi *finalizeTadpole(3)* menyatakan bahwa simpul yang dilabel berikutnya adalah simpul v_3 yang terdapat pada graf lingkaran. Karena simpul ini merupakan simpul terakhir pada graf C_3 , maka proses berlanjut ke baris pertama pada Algoritma 4.c. Label untuk simpul v_3 diperoleh dengan cara $k - \lambda(e_3) - \lambda(e_2)$, sehingga diperoleh $\lambda(v_3) = -1$ (Gambar 3.26 (a)). Karena label -1 ini tidak tersedia, maka proses selanjutnya adalah menuju baris ke-13 pada Algoritma 4.c, yaitu keluar dari fungsi *finalizeTadpole(3)* untuk melakukan *backtracking* ke fungsi *finalizeTadpole(2)*. Nyatakan *avail[12] = true*, yang berarti label 12 tersedia kembali. Label 12 ini merupakan label yang digunakan untuk melabel elemen yang dilabel sebelum melabel simpul v_3 , yaitu busur e_2 . Dengan kata lain, busur e_2 gagal dilabel oleh 12 (Gambar 3.26 (b)). Kemudian, nyatakan *avail[3] = true*, yang berarti label 3 tersedia kembali. Label 3 ini merupakan label yang digunakan untuk melabel elemen yang dilabel sebelum melabel busur e_2 , yaitu simpul v_2 . Dengan kata lain, simpul v_2 gagal dilabel oleh 3 (Gambar 3.26 (c)). Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 4, 5, 6, 7, 9, 10, dan 11.



Gambar 3. 27 Proses pada *finalizeTadpole(2)* dan *finalizeTadpole(3)* untuk membentuk PTSA $T_{3,3}$ dengan $k = 17$

Proses selanjutnya adalah menuju baris ke- 6 pada Algoritma 4.c., yaitu simpul v_2 akan dilabel kembali dengan label lain yang masih tersedia. Misalkan pilih 12, maka $\lambda(v_2) = 12$ dan nyatakan $avail[12] = false$ (Gambar 3.27 (a)). Label untuk busur e_2 dapat ditentukan, yaitu $\lambda(e_2) = k - \lambda(v_2) - \lambda(e_1) = 3$. Karena label ini masih tersedia, maka nyatakan $avail[3] = false$ (Gambar 3.27 (b)). Fungsi *finalizeTadpole* ini kemudian akan memanggil dirinya sendiri dengan parameter $t+1 = 2+1 = 3$. Sampai sejauh ini, label yang telah digunakan untuk melabel adalah 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, dan 12. Karena elemen berikutnya yang akan dilabel merupakan simpul terakhir pada graf C_3 , maka proses berlanjut ke baris pertama pada Algoritma 4.c. Label untuk simpul v_3 diperoleh dengan cara $k - \lambda(e_3) - \lambda(e_2)$, sehingga diperoleh $\lambda(v_3) = 8$ (Gambar 3.27 (c)). Karena label ini merupakan satu-satunya label yang belum digunakan, maka terbentuklah satu PTSA $T_{3,3}$ dengan nilai $k = 17$, yaitu dengan $\lambda(v_1) = 4$, $\lambda(v_2) = 12$, $\lambda(v_3) = 8$, $\lambda(v_4) = 11$, $\lambda(v_5) = 9$, $\lambda(v_6) = 10$, $\lambda(e_1) = 2$, $\lambda(e_2) = 3$, $\lambda(e_3) = 6$, $\lambda(e_4) = 5$, $\lambda(e_5) = 1$, $\lambda(e_6) = 7$. Label-label inilah yang akan dicetak sebagai *output*. Walaupun sudah dihasilkan suatu PTSA $T_{3,3}$, proses pada algoritma ini belum berhenti. Proses tetap dilakukan secara *backtracking* sampai dihasilkan semua PTSA yang tidak isomorfik untuk graf matahari $T_{3,3}$ dengan nilai $k = 17$.

BAB 4 IMPLEMENTASI DAN SIMULASI ALGORITMA PTSA

Pada Bab 3 telah diberikan algoritma-algoritma PTSA untuk graf lingkaran C_n , graf matahari $C_n \odot \bar{K}_1$, dan graf kecebong $T_{m,n}$. Selanjutnya, pada Subbab ini akan dijelaskan mengenai implementasi algoritma-algoritma PTSA tersebut dengan menggunakan program MATLAB beserta simulasinya. Program dijalankan pada *Note Book* dengan prosesor *Intel Dual Core @ 2.00 GHz*, memori 1.87 GB, dan sistem operasi *Microsoft Windows XP Professional*.

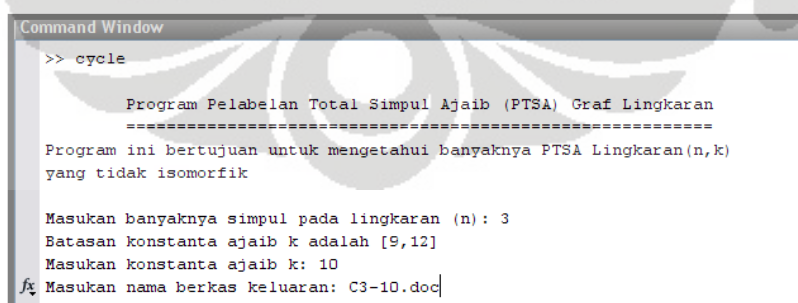
Pada implementasi dengan menggunakan program yang akan diberikan, himpunan label yang tersedia untuk melabel elemen pada graf ditandai dengan L . Pada algoritma, pengertian dari L ini sama dengan *avail*. Namun, jika *avail* merupakan suatu *array* berukuran $|V|+|E|$ yang berisikan nilai *true* atau *false*, L disini tidak demikian. L pada program merupakan suatu *array* berukuran $|V|+|E|$ yang berisi label-label untuk melabel elemen pada graf, yaitu $1, 2, \dots, |V|+|E|$. Setiap label-label ini digunakan tepat satu kali. Jika label sudah digunakan untuk melabel elemen x , maka tandai $L(x) = 0$. Jika ternyata label yang digunakan bukan merupakan label yang tepat untuk melabel elemen x , maka tandai $L(x) = x$, yang berarti label x tersedia kembali.

Program-program PTSA untuk lingkaran C_n , graf matahari $C_n \odot \bar{K}_1$, dan graf kecebong $T_{m,n}$ dapat menghasilkan semua PTSA tidak isomorfik yang mungkin untuk setiap nilai n (dan m untuk graf kecebong $T_{m,n}$) dan k yang diberikan. Namun, karena PTSA tidak isomorfik yang dihasilkan cukup banyak, maka pada simulasi yang dilakukan, yang menjadi perhatian hanyalah banyaknya PTSA tidak isomorfik yang terjadi untuk graf-graf terkait dengan nilai k berdasarkan (3.1), (3.4), dan (3.8). Dengan sendirinya, menggunakan program-program ini dapat juga diketahui ada atau tidaknya PTSA dari masing-masing graf dengan nilai n (dan m untuk graf kecebong $T_{m,n}$) dan k yang diberikan. Pada subbab selanjutnya akan diberikan implementasi dari algoritma PTSA untuk graf lingkaran beserta simulasinya.

4.1 Implementasi dan Simulasi Algoritma PTSA untuk Graf Lingkaran

Algoritma PTSA untuk graf lingkaran C_n telah diberikan pada Subbab 3.1. Berikut akan dijelaskan mengenai implementasi dari algoritma ini dalam program. Algoritma PTSA untuk graf lingkaran secara keseluruhan akan dijalankan oleh fungsi `cycle`. Fungsi `cycle` akan memanggil fungsi `initializecycle`. Fungsi `initializecycle` ini merupakan implementasi dari fungsi inisialisasi yang diberikan pada Algoritma 2.a. Kemudian, fungsi `initializecycle` akan memanggil fungsi `extendcycle`. Fungsi `extendcycle` ini merupakan implementasi dari fungsi perluasan yang diberikan pada Algoritma 2.b. Untuk lebih jelasnya, *listing* program PTSA untuk graf lingkaran ini dapat dilihat pada Lampiran 2 (CD).

Pemanggilan program dilakukan dengan mengetik “`cycle`” pada *Command Window*. Pengguna (*user*) akan diminta untuk memasukkan nilai n dan k yang diinginkan, yang menyatakan banyaknya simpul pada graf lingkaran C_n dan nilai konstanta ajaib yang diinginkan sesuai dengan batasan yang diperoleh berdasarkan (3.1). Sebagai keluaran dari penggunaan program ini, akan diperoleh label-label simpul dan busur yang membentuk suatu PTSA pada graf C_n untuk nilai n dan k yang dimasukkan pengguna. Hasil ini akan disimpan pada suatu berkas (*file*) keluaran. Karena hasil ini disimpan dalam suatu berkas, maka pengguna juga akan diminta untuk memasukkan nama berkas keluaran (*output file*) yang diinginkan dalam format `.doc`. Jadi, pada saat memanggil program, tampilan yang pertama kali muncul pada *Command Window* adalah seperti Gambar 4.1.



```

Command Window
>> cycle

      Program Pelabelan Total Simpul Ajaib (PTSA) Graf Lingkaran
      =====
Program ini bertujuan untuk mengetahui banyaknya PTSA Lingkaran(n,k)
yang tidak isomorfik

Masukan banyaknya simpul pada lingkaran (n): 3
Batasan konstanta ajaib k adalah [9,12]
Masukan konstanta ajaib k: 10
fx Masukan nama berkas keluaran: C3-10.doc
  
```

Gambar 4.1 Tampilan *Command Window* awal

Program akan berhenti ketika sudah dihasilkan semua PTSA tidak isomorfik untuk graf lingkaran. Selain dihasilkan label-label dari semua PTSA yang

mungkin, program ini juga memberikan banyaknya PTSA tidak isomorfik yang terjadi. Sebagai contoh penggunaan program PTSA ini, akan dilakukan simulasi untuk graf lingkaran yang digunakan sebagai contoh dari penggunaan algoritma yang diberikan pada Subbab 3.1, yaitu graf C_3 dengan nilai $k = 10$. Hasil yang diperoleh dari penggunaan program pada contoh ini akan disimpan dalam berkas dengan nama C3-10.doc. Tampilan masukan dan keluaran yang dihasilkan diberikan pada Gambar 4.2.

```

Command Window
>> cycle

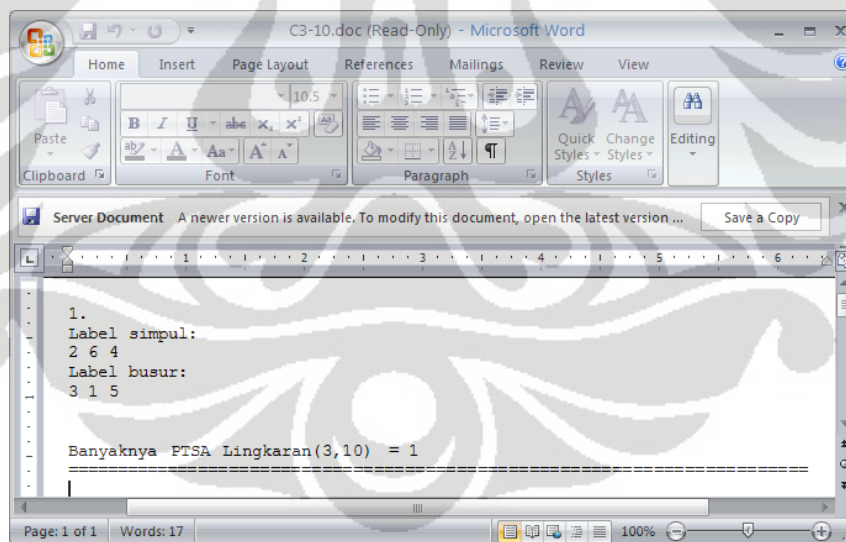
      Program Pelabelan Total Simpul Ajaib (PTSA) Graf Lingkaran
      =====
Program ini bertujuan untuk mengetahui banyaknya PTSA Lingkaran(n,k)
yang tidak isomorfik

Masukan banyaknya simpul pada lingkaran (n): 3
Batasan konstanta ajaib k adalah [9,12]
Masukan konstanta ajaib k: 10
Masukan nama berkas keluaran: C3-10.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
2
Bye...
fx >>

```

(a)

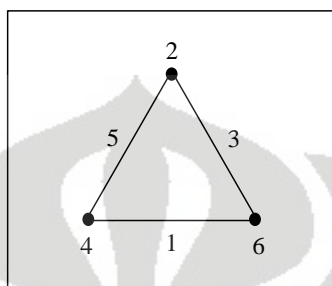


(b)

Gambar 4.2 Tampilan masukan (a) dan keluaran (b) untuk PTSA C_3 dengan $k = 10$

Dari Gambar 4.2 (b) dapat terlihat bahwa hanya terdapat satu PTSA untuk graf lingkaran C_3 dengan $k = 10$. Graf C_3 dengan PTSA tersebut memiliki label 2

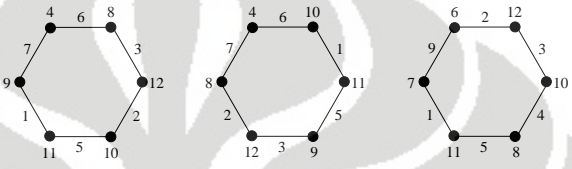
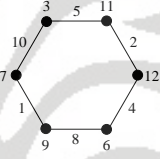
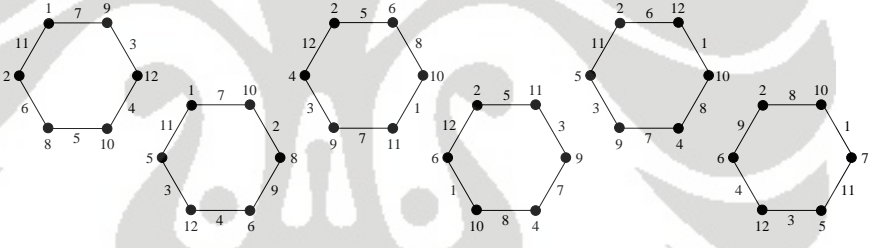
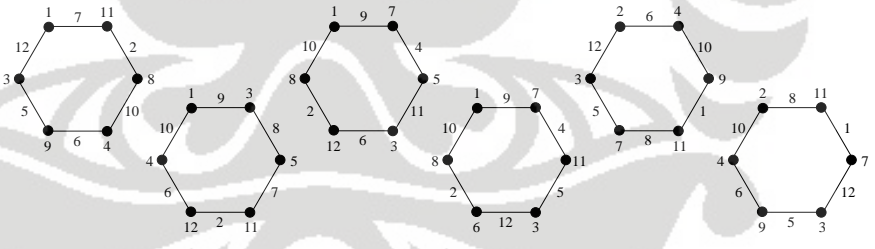
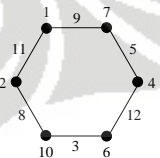
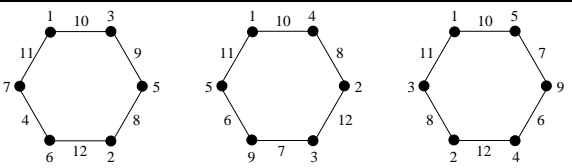
untuk simpul v_1 , 6 untuk simpul v_2 , 4 untuk simpul v_3 , 3 untuk busur e_1 , 1 untuk busur e_2 , dan 5 untuk busur e_3 . Untuk lebih jelasnya dapat dilihat pada Gambar 4.3 yang memberikan graf lingkaran C_3 yang memiliki PTSA dengan label-label yang diberikan pada Gambar 4.2 (b).



Gambar 4.3 PTSA C_3 dengan $k = 10$

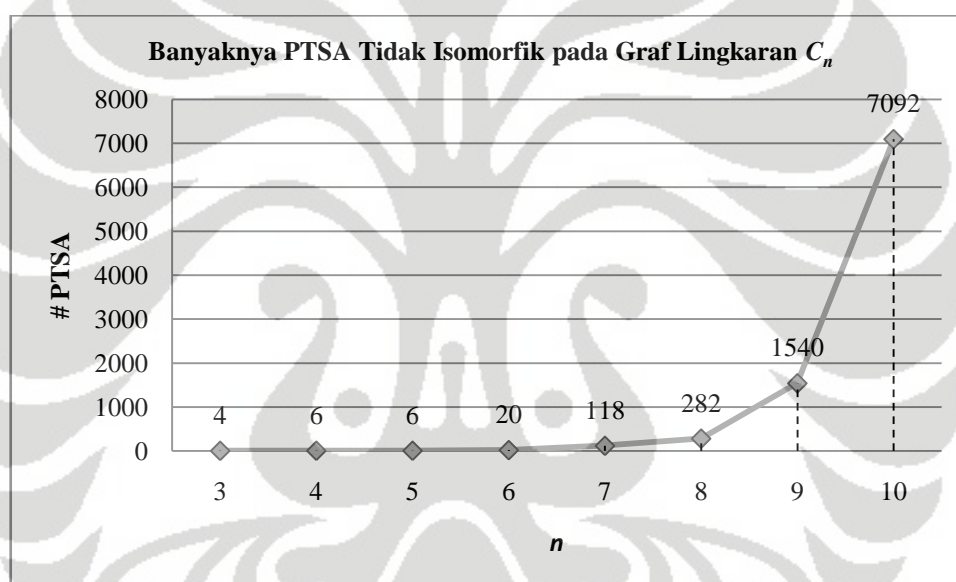
Sebagai contoh lainnya dari penggunaan program PTSA graf lingkaran ini, akan dilakukan simulasi pada graf C_6 untuk semua nilai k yang mungkin, yang diperoleh berdasarkan (3.1), yaitu 17, 18, ..., 22. Tampilan masukan dan keluaran dari penggunaan program ini dapat dilihat pada Lampiran 1. Pada Gambar 4.4 diberikan graf C_6 yang memiliki PTSA dengan label-label berdasarkan Lampiran 2. Dari Gambar 4.4 terlihat hal yang menarik. Banyaknya PTSA C_6 untuk $k = 17$ sama dengan banyaknya PTSA C_6 untuk $k = 22$, untuk $k = 18$ sama dengan untuk $k = 21$, dan untuk $k = 19$ sama dengan untuk $k = 20$. Ternyata, pasangan PTSA C_6 dengan nilai-nilai k ini memenuhi perumusan yang diberikan pada Definisi 2.2 mengenai pelabelan dual. Sebagai contohnya, perhatikan pasangan PTSA C_6 dengan nilai $k = 18$ dan $k = 21$ pada Gambar 4.4 (b) dan (e). Misalkan label untuk simpul v_1 ($\lambda(v_1)$) dan busur e_1 ($\lambda(e_1)$) untuk PTSA C_6 dengan $k = 18$ pada Gambar 4.4 (b) masing-masing adalah 3 dan 5. Karena pada graf C_6 , $|V| = |E| = 6$, maka untuk memperoleh PTSA dualnya, label simpul v_1 ($\lambda'(v_1)$) dan busur e_1 ($\lambda'(e_1)$) untuk PTSA dual tersebut dihasilkan berdasarkan perumusan yang diberikan pada Definisi 2.2. Berdasarkan Definisi tersebut diperoleh $\lambda'(v_1) = 6 + 6 + 1 - 3 = 10$ dan $\lambda'(e_1) = 6 + 6 + 1 - 5 = 8$. Cara yang sama dilakukan untuk label simpul dan label busur yang lain. Label-label untuk simpul dan busur yang diperoleh ini akan sama dengan label-label untuk simpul dan busur pada Gambar 4.4 (e) apabila dilakukan rotasi terhadap label-label tersebut. Ini berarti, PTSA C_6 dengan $k = 18$ dan

$k = 21$ pada Gambar 4.4 (b) dan (e) merupakan PTSA yang saling dual. Hal ini juga berlaku untuk pasangan PTSA C_6 dengan nilai-nilai k yang lain. Jadi, terlihat bahwa PTSA C_6 dengan $k = 17$ merupakan dual dari PTSA C_6 dengan $k = 22$, PTSA C_6 dengan $k = 18$ merupakan dual dari PTSA C_6 dengan $k = 21$, dan PTSA C_6 dengan $k = 19$ merupakan dual dari PTSA C_6 dengan $k = 20$. Artinya, hasil dari program sesuai dengan teori yang ada, yaitu PTSA graf teratur memiliki dual yang juga merupakan PTSA (Teorema 2.1).

(a)	$k = 17$	
(b)	$k = 18$	
(c)	$k = 19$	
(d)	$k = 20$	
(e)	$k = 21$	
(f)	$k = 22$	

Gambar 4.4 PTSA-PTSA tidak isomorfik pada graf C_6

Berdasarkan Tabel 4.1 dapat terlihat bahwa hasil tersebut sesuai dengan hasil penelitian yang telah dipublikasikan, yang terdapat pada makalah MacDougall *et al.* (2002), yaitu graf lingkaran C_n memiliki suatu PTSA untuk nilai $n \geq 3$. Pada makalah tersebut juga terdapat keterangan bahwa graf C_5 tidak memiliki PTSA dengan $k = 15$ dan $k = 18$. Hasil-hasil yang diperoleh ini sama dengan hasil yang terdapat pada makalah Baker dan Sawada (2008) dan Macdougall *et al.* (2002), yang masing-masing memberikan banyaknya PTSA tidak isomorfik yang terjadi pada graf lingkaran. Jadi, berdasarkan Tabel 4.1 dapat disimpulkan bahwa graf lingkaran C_n dengan $n = 3$ s.d. 10 memiliki PTSA untuk setiap nilai k yang mungkin kecuali C_5 dengan $k = 15$ dan $k = 18$.



Gambar 4.5 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf C_n seiring dengan bertambahnya nilai n

Pada Gambar 4.5 diberikan grafik yang menunjukkan perubahan banyaknya PTSA tidak isomorfik yang terjadi pada graf lingkaran C_n seiring dengan bertambahnya nilai n . Dari gambar tersebut terlihat bahwa kecenderungan pertambahan banyaknya PTSA tidak isomorfik yang terjadi sesuai dengan grafik fungsi eksponensial. Jadi, terdapat peningkatan yang signifikan untuk banyaknya PTSA seiring dengan bertambahnya nilai n . Hal ini dapat dilihat untuk 3 nilai n terakhir, yaitu $n = 8, 9, 10$. Oleh karena pertambahan banyaknya PTSA meningkat secara eksponensial, maka untuk nilai n yang lebih besar diperlukan *running time* yang

cukup lama. Simulasi untuk nilai n yang lebih besar dari 10 belum dapat dilakukan karena keterbatasan waktu.

Pada subbab selanjutnya akan diberikan implementasi dari algoritma PTSA pada graf matahari beserta simulasinya.

4.2 Implementasi dan Simulasi Algoritma PTSA untuk Graf Matahari

Pada Subbab 3.2 telah dijelaskan mengenai algoritma PTSA untuk graf matahari $C_n \odot \bar{K}_1$. Algoritma-algoritma tersebut akan diimplementasikan dalam bentuk program. Algoritma PTSA graf matahari secara keseluruhan akan dijalankan oleh fungsi `sun`. Fungsi `sun` akan memanggil fungsi `initializesun` yang merupakan implementasi dari fungsi inisialisasi yang diberikan pada Algoritma 3.a. Kemudian, fungsi `initializesun` akan memanggil fungsi `extendsun` yang merupakan implementasi dari fungsi perluasan yang diberikan pada Algoritma 3.b. Untuk lebih jelasnya, *listing* program PTSA untuk graf matahari ini dapat dilihat pada Lampiran 3 (CD).

Pemanggilan program dilakukan dengan mengetik “`sun`” pada *Command Window*. Sama seperti program PTSA untuk graf lingkaran, pada program PTSA untuk graf matahari ini pengguna akan diminta untuk memasukkan nilai n dan k yang diinginkan, yang menyatakan ukuran graf matahari $C_n \odot \bar{K}_1$ dan nilai konstanta ajaib yang diinginkan sesuai dengan batasan yang diperoleh berdasarkan (3.4). Selain itu, pengguna juga diminta untuk memasukkan nama berkas keluaran yang diinginkan dengan format `.doc`.

Seperti halnya keluaran pada program PTSA graf lingkaran, keluaran pada program PTSA graf matahari juga berupa label-label simpul dan busur. Perlu diingat bahwa simpul dan busur pada graf matahari terbagi menjadi dua jenis, yaitu simpul dan busur dalam serta simpul dan busur luar. Program akan berhenti jika sudah dihasilkan semua PTSA tidak isomorfik untuk nilai n dan k yang dimasukkan pengguna. Banyaknya PTSA tidak isomorfik yang terjadi juga menjadi keluaran pada program ini. Sebagai contoh penggunaan program, akan dilakukan simulasi untuk graf matahari yang digunakan sebagai contoh dari penggunaan algoritma yang diberikan pada Subbab 3.2, yaitu graf $C_4 \odot \bar{K}_1$ dengan $k = 25$. Ha-

sil yang diperoleh dari penggunaan program pada contoh ini akan disimpan dengan nama C4K1-25.doc. Tampilan masukan dan keluaran yang dihasilkan diberikan pada Gambar 4.6.

```

Command Window
>> sun

      Program Pelabelan Total Simpul Ajaib (PTSA) Graf Matahari
      =====
Program ini bertujuan untuk mengetahui banyaknya PTSA Matahari (n,k)
yang tidak isomorfik

Masukan banyaknya simpul pada matahari: 4
Batasan konstanta ajaib k adalah [22,25]
Masukan bobot simpul: 25
Masukan nama berkas keluaran: C4K1-25.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
2
Bye...
fx >> |

```

(a)

```

Screen 1 of 2
document, open the latest version or save a copy. Save a Copy

1.
Label simpul:
1 7 8 5 9 13 15 14
Label busur:
2 4 3 6 16 12 10 11

2.
Label simpul:
1 8 7 5 9 13 14 15
Label busur:
2 3 4 6 16 12 11 10

3.
Label simpul:
1 6 8 4 10 13 16 14
Label busur:
2 5 3 7 15 12 9 11

4.
Label simpul:
1 8 6 4 10 13 14 16
Label busur:
2 3 5 7 15 12 11 9

5.
Label simpul:
1 8 5 6 10 14 12 16
Label busur:
2 4 3 7 15 11 13 9

6.
Label simpul:
1 5 7 4 11 13 16 15

Screen 2 of 2
document, open the latest version or save a copy. Save a Copy

Label busur:
2 6 3 8 14 12 9 10

7.
Label simpul:
1 6 7 3 11 13 16 15
Label busur:
2 5 4 8 14 12 9 10

8.
Label simpul:
1 7 6 3 11 13 15 16
Label busur:
2 4 5 8 14 12 10 9

9.
Label simpul:
1 7 4 5 11 15 13 16
Label busur:
2 6 3 8 14 10 12 9

10.
Label simpul:
1 7 8 6 9 12 14 15
Label busur:
3 2 4 5 16 13 11 10

Banyaknya PTSA Matahari(4,25) = 118
=====

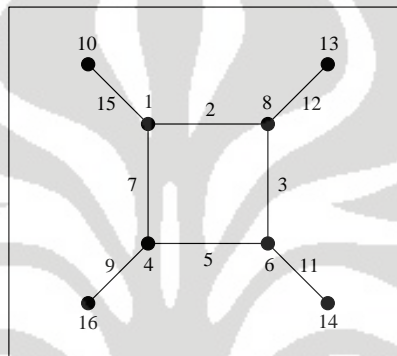
```

(b)

Gambar 4.6 Tampilan masukan (a) dan keluaran (b) untuk PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$

Dari Gambar 4.6 (b) dapat terlihat bahwa terdapat 118 PTSA tidak isomorfik untuk graf $C_4 \odot \bar{K}_1$ dengan $k = 25$. Seperti yang telah disebutkan pada awal bab, karena hasil PTSA untuk nilai k ini cukup banyak, hanya 10 PTSA yang di-

tampilkan sebagai keluaran. PTSA $C_4 \odot \bar{K}_1$ yang digunakan sebagai contoh penggunaan algoritma pada Subbab 3.2 adalah PTSA nomor 4. Graf $C_4 \odot \bar{K}_1$ tersebut memiliki label simpul dalam dan simpul luar berturut-turut adalah 1, 8, 6, 4, 10, 13, 14, 16. Sedangkan untuk label busur dalam dan busur luar berturut-turut adalah 2, 3, 5, 7, 15, 12, 11, 9. Pada Gambar 4.7 diberikan gambar graf $C_4 \odot \bar{K}_1$ yang memiliki PTSA dengan label-label yang diberikan pada Gambar 4.6 (b) nomor 4.



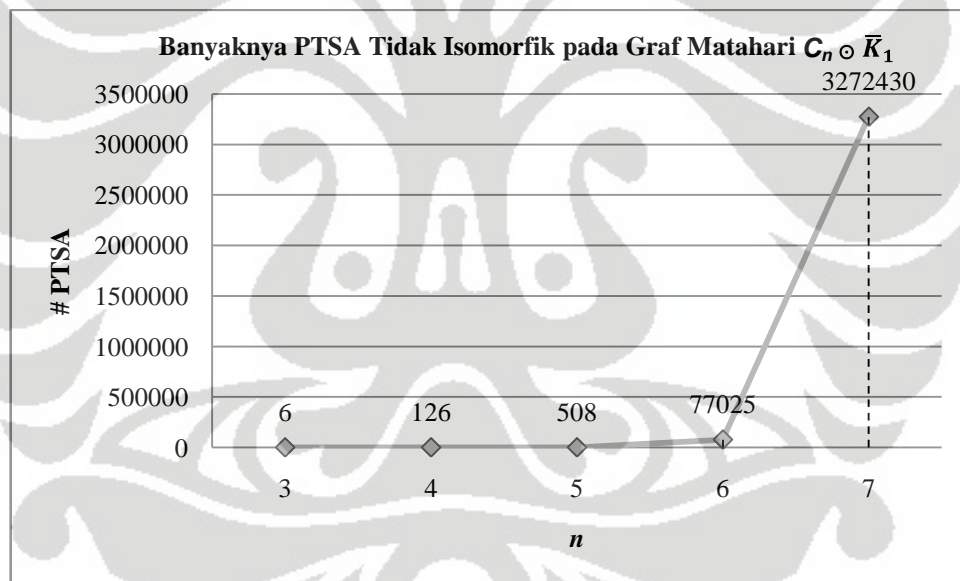
Gambar 4.7 PTSA $C_4 \odot \bar{K}_1$ dengan $k = 25$

Tabel 4.2 Banyaknya PTSA tidak isomorfik untuk graf matahari $C_n \odot \bar{K}_1$ dengan $n = 3$ s.d. 7 berdasarkan $5n + 2 \leq k \leq 6n + 1$

$C_3 \odot \bar{K}_1$		$C_4 \odot \bar{K}_1$		$C_5 \odot \bar{K}_1$		$C_6 \odot \bar{K}_1$		$C_7 \odot \bar{K}_1$	
k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA
17	0	22	0	27	0	32	0	37	0
18	1	23	2	28	12	33	147	38	1824
19	5	24	6	29	93	34	580	39	10844
		25	118	30	190	35	3554	40	31367
				31	213	36	7966	41	190451
						37	64778	42	328510
								43	2709434
Total : 6		Total : 126		Total : 508		Total : 77025		Total : 3272430	

Pada Tabel 4.2 diberikan hasil simulasi dari program PTSA untuk graf matahari $C_3 \odot \bar{K}_1$ sampai $C_7 \odot \bar{K}_1$ yang berupa banyaknya PTSA tidak isomorfik yang terjadi dengan nilai k berdasarkan batasan yang diberikan pada (3.4). Tidak seperti pada graf lingkaran, pada graf matahari tidak terdapat PTSA dual karena graf matahari bukan merupakan graf teratur. Berdasarkan Tabel 4.2 dapat terlihat

bahwa selalu terdapat PTSA untuk graf matahari $C_n \odot \bar{K}_1$ dengan nilai $k = 6n+1$. Hasil ini sesuai dengan hasil penelitian yang telah dipublikasikan, yang terdapat pada Teorema 2.3 dan 2.4, yaitu mengenai gabungan graf matahari $(C_{t_1} \odot \bar{K}_1) \cup (C_{t_2} \odot \bar{K}_1) \cup \dots \cup (C_{t_n} \odot \bar{K}_1)$ dengan $t_i \geq 3$ untuk setiap $i = 1, 2, \dots, n$ dan $n \geq 1$, baik yang saling isomorfik maupun yang tidak harus saling isomorfik, memiliki PTSA dengan nilai $k = 6 \sum_{k=1}^n t_k + 1$. Berdasarkan kedua teorema ini, jika $t_i = 1$, maka diperoleh nilai $k = 6n+1$. Selain dengan nilai k ini, ternyata diperoleh juga PTSA untuk graf matahari dengan nilai k yang lain, sesuai dengan (3.4). Akan tetapi, dapat terlihat pada Tabel 4.2 bahwa selalu tidak terdapat suatu PTSA untuk graf matahari $C_n \odot \bar{K}_1$ dengan nilai $k = 5n+2$. Oleh karena itu, perlu dilakukan penelitian lebih lanjut untuk membuktikan hal ini secara analitik atau mencari cara lain untuk memperketat batasan batasan nilai k pada graf matahari.



Gambar 4.8 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $C_n \odot \bar{K}_1$ seiring dengan bertambahnya nilai n

Pada Gambar 4.8 diberikan grafik yang menunjukkan perubahan banyaknya PTSA tidak isomorfik yang terjadi pada graf lingkaran $C_n \odot \bar{K}_1$ seiring dengan bertambahnya nilai n . Seperti pada grafik banyaknya PTSA untuk graf lingkaran, terdapat peningkatan yang signifikan untuk banyaknya PTSA tidak isomorfik pada graf matahari apabila nilai n bertambah. Kecenderungan pertambahan banyaknya PTSA ini sesuai dengan grafik fungsi eksponensial sedemikian sehingga un-

tuk nilai n yang lebih besar diperlukan *running time* yang lebih lama. Simulasi untuk nilai n yang lebih besar dari 7 belum dapat dilakukan karena menghabiskan waktu lebih lama.

Perlu diperhatikan bahwa pada graf matahari $C_n \odot \bar{K}_1$, $|V| = |E| = 2n$, sedangkan pada graf lingkaran C_n , $|V| = |E| = n$, sehingga dapat dipahami bahwa untuk nilai n yang sama, banyaknya PTSA tidak isomorfik dari graf matahari jauh lebih banyak daripada graf lingkaran. Oleh karena itu, waktu yang diperlukan untuk mencari semua PTSA tidak isomorfik dari graf matahari lebih lama bila dibandingkan dengan graf lingkaran. Pada subbab selanjutnya akan diberikan implementasi dari algoritma PTSA pada graf kecebong beserta simulasinya.

4.3 Implementasi dan Simulasi Algoritma PTSA untuk Graf Kecebong

Algoritma PTSA untuk graf kecebong $T_{m,n}$ telah diberikan pada Subbab 3.3. Seperti Algoritma PTSA untuk kedua graf yang telah dibahas sebelumnya, algoritma PTSA untuk graf kecebong ini juga akan diimplementasikan dalam bentuk program. Algoritma PTSA ini secara keseluruhan akan dijalankan oleh fungsi *tadpole*. Fungsi *tadpole* akan memanggil fungsi *initializetadpole* yang merupakan implementasi dari fungsi inisialisasi yang diberikan pada Algoritma 4.a. Kemudian, fungsi *initializetadpole* akan memanggil fungsi *extendtadpole* yang merupakan implementasi dari fungsi perluasan yang diberikan pada Algoritma 4.b. Pada fungsi *extendtadpole* ini juga terjadi pemanggilan fungsi, yaitu fungsi *finalizetadpole*, yang merupakan implementasi dari fungsi finalisasi yang diberikan pada Algoritma 4.c. Untuk lebih jelasnya, *listing* program PTSA untuk graf kecebong ini dapat dilihat pada Lampiran 4 (CD).

Pemanggilan program dilakukan dengan mengetik “*tadpole*” pada *Command Window*. Karena graf kecebong terdiri dari graf lingkaran C_m dan graf lintasan P_n , maka pada program PTSA ini pengguna akan diminta untuk memasukkan nilai m , n dan k yang diinginkan. Nilai k ini menyatakan konstanta ajaib yang diperoleh berdasarkan (3.8). Pengguna juga diminta untuk memasukkan nama berkas keluaran yang diinginkan dengan format *.doc*.

```

Command Window
>> tadpole

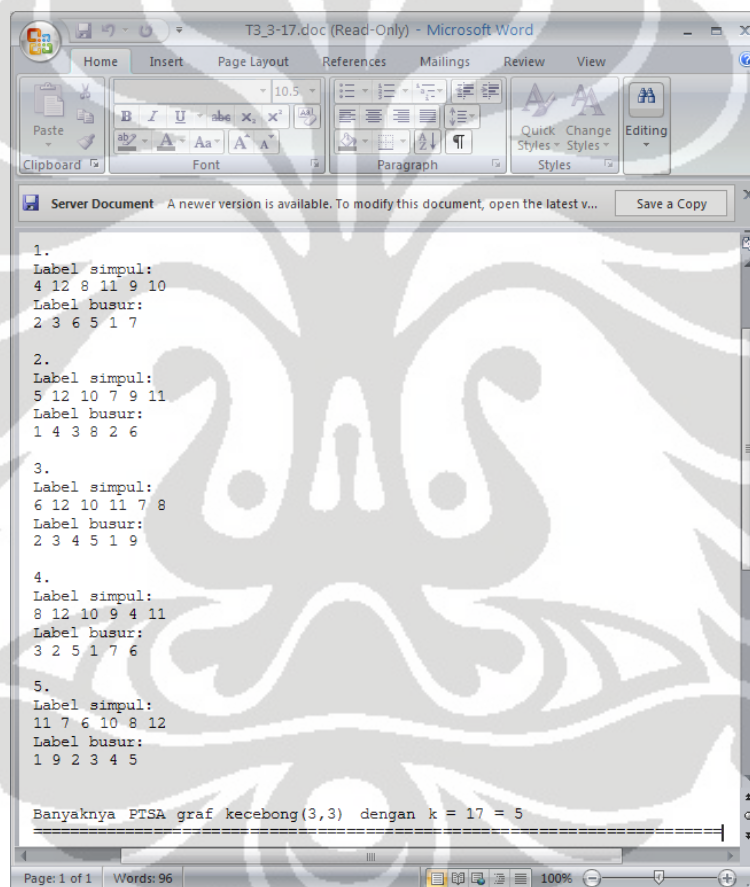
      Program Pelabelan Total Simpul Ajaib (PTSA) Graf Kecebong
      -----
Program ini bertujuan untuk mengetahui banyaknya PTSA graf kecebong(m,n)
yang tidak isomorfik

Masukan m: 3
Masukan n: 3
Batasan konstanta ajaib k adalah [17,22]
Masukan konstanta ajaib k: 17
Masukan nama berkas keluaran: T3_3-17.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
2
Bye...
fx >> |

```

(a)



```

T3_3-17.doc (Read-Only) - Microsoft Word
Home Insert Page Layout References Mailings Review View
Clipboard Font Paragraph Styles Editing
Server Document A newer version is available. To modify this document, open the latest v... Save a Copy

1.
Label simpul:
4 12 8 11 9 10
Label busur:
2 3 6 5 1 7

2.
Label simpul:
5 12 10 7 9 11
Label busur:
1 4 3 8 2 6

3.
Label simpul:
6 12 10 11 7 8
Label busur:
2 3 4 5 1 9

4.
Label simpul:
8 12 10 9 4 11
Label busur:
3 2 5 1 7 6

5.
Label simpul:
11 7 6 10 8 12
Label busur:
1 9 2 3 4 5

Banyaknya PTSA graf kecebong(3,3) dengan k = 17 = 5

Page: 1 of 1 Words: 96 100%

```

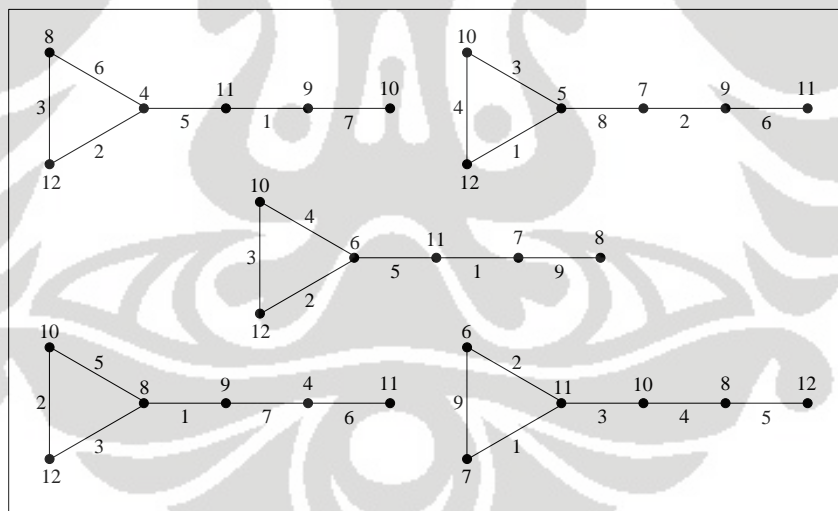
(b)

Gambar 4.9 Tampilan masukan (a) dan keluaran (b) untuk PTSA $T_{3,3}$ dengan $k = 17$

Seperti halnya keluaran untuk program PTSA pada graf-graf yang telah dibahas sebelumnya, keluaran untuk program PTSA graf kecebong juga berupa label-label simpul dan label-label busur yang membentuk suatu PTSA pada graf ter-

sebut. Program akan berhenti jika sudah dihasilkan semua PTSA tidak isomorfik untuk graf kecebong. Banyaknya PTSA tidak isomorfik yang terjadi juga menjadi keluaran pada program ini. Sebagai contoh penggunaan program PTSA ini, akan dilakukan simulasi untuk graf kecebong yang digunakan sebagai contoh dari penggunaan algoritma yang diberikan pada Subbab 3.3, yaitu graf $T_{3,3}$ dengan nilai $k = 17$. Hasil yang diperoleh dari penggunaan program pada contoh ini akan disimpan dengan nama T3_3-17.doc. Tampilan masukan dan keluaran yang dihasilkan diberikan pada Gambar 4.9.

Dari Gambar 4.9 (b) dapat terlihat bahwa terdapat 5 PTSA untuk graf kecebong $T_{3,3}$ dengan $k = 17$. PTSA $T_{3,3}$ yang digunakan sebagai contoh penggunaan algoritma pada Subbab 3.3 adalah PTSA yang pertama dengan label simpul berturut-turut adalah 4, 12, 8, 11, 9, 10 dan label busur berturut-turut adalah 2, 3, 6, 5, 1, 7. Untuk lebih jelasnya dapat dilihat pada Gambar 4.10 yang memberikan semua PTSA tidak isomorfik pada $T_{3,3}$ dengan label-label yang diberikan pada Gambar 4.9 (b).



Gambar 4.10 5 PTSA tidak isomorfik pada graf $T_{3,3}$ dengan $k = 17$

Pada Tabel 4.3 sampai 4.7 diberikan hasil simulasi dari program PTSA untuk graf kecebong berdasarkan nilai k yang diperoleh pada (3.8). Simulasi baru dapat dilakukan untuk nilai $3 \leq m \leq 7$ dan $1 \leq n \leq 5$. Dari hasil simulasi ini dapat terlihat bahwa dengan batasan nilai k yang telah diperketat, yang diberikan pada (3.8) ternyata masih terdapat graf kecebong $T_{m,n}$ yang tidak memiliki PTSA, se-

perti pada $T_{3,1}$ untuk $k = 12$ dan 13 , $T_{3,2}$ untuk $k = 15$ dan 18 , $T_{3,3}$ untuk $k = 22$, $T_{3,5}$ untuk $k = 29$, dan sebagainya. Data-data ini dapat menjadi bahan penelitian lebih jauh untuk membuktikan secara analitik ketiadaan PTSA graf kecebong $T_{m,n}$ untuk nilai m , n , dan k yang telah disebutkan sebelumnya. Seperti pada graf matahari, pada graf kecebong juga tidak terdapat PTSA dual karena graf kecebong bukan merupakan graf teratur.

Tabel 4.3 Banyaknya PTSA tidak isomorfik untuk graf $T_{3,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)}\right)$

T(3,1)		T(3,2)		T(3,3)		T(3,4)		T(3,5)	
k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA
12	0	14	1	17	5	19	16	22	45
13	0	15	0	18	5	20	23	23	61
14	1	16	1	19	7	21	38	24	160
		17	2	20	6	22	64	25	185
		18	0	21	3	23	56	26	141
				22	0	24	27	27	110
						25	10	28	22
								29	0
Total : 1		Total : 4		Total : 26		Total : 234		Total : 724	

Tabel 4.4 Banyaknya PTSA tidak isomorfik untuk graf $T_{4,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)}\right)$

T(4,1)		T(4,2)		T(4,3)		T(4,4)		T(4,5)	
k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA
14	3	17	6	19	11	22	52	24	105
15	2	18	2	20	16	23	79	25	186
16	3	19	10	21	27	24	151	26	463
17	4	20	4	22	29	25	201	27	1021
18	0	21	2	23	25	26	169	28	958
		22	0	24	9	27	142	29	1082
				25	0	28	18	30	425
						29	0	31	257
								32	15
Total : 12		Total : 24		Total : 117		Total : 812		Total : 4512	

Sejauh yang dapat dicari dalam literatur, belum ada penelitian mengenai eksistensi PTSA pada graf kecebong, sehingga berdasarkan simulasi dapat disimpulkan bahwa graf kecebong $T_{m,n}$ memiliki PTSA untuk nilai m , n , dan k seperti

Pada Tabel 4.8. Setiap sel pada Tabel ini menunjukkan nilai k dari graf kecebong yang memiliki PTSA.

Tabel 4.5 Banyaknya PTSA tidak isomorfik untuk graf $T_{5,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)}\right)$

T(5,1)		T(5,2)		T(5,3)		T(5,4)		T(5,5)	
k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA
17	2	19	24	22	26	24	112	27	659
18	19	20	20	23	53	25	197	28	1101
19	10	21	23	24	106	26	477	29	3282
20	9	22	49	25	166	27	796	30	4408
21	3	23	71	26	98	28	909	31	6213
22	0	24	10	27	116	29	1061	32	5260
		25	9	28	4	30	647	33	3850
				29	0	31	240	34	1529
						32	15	35	342
								36	0
Total : 43		Total : 206		Total : 569		Total : 4454		Total : 26644	

Tabel 4.6 Banyaknya PTSA tidak isomorfik untuk graf $T_{6,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)}\right)$

T(6,1)		T(6,2)		T(6,3)		T(6,4)		T(6,5)	
k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA
19	9	22	56	24	118	27	539	29	1459
20	15	23	99	25	184	28	1223	30	3669
21	42	24	220	26	431	29	2933	31	9210
22	39	25	257	27	843	30	3988	32	17630
23	34	26	209	28	900	31	5423	33	25835
24	17	27	178	29	985	32	5038	34	30850
25	0	28	38	30	637	33	3500	35	32018
		29	0	31	205	34	1584	36	16610
				32	15	35	357	37	10060
						36	0	38	2846
								39	346
								40	0
Total : 156		Total : 1057		Total : 4318		Total : 24585		Total : 150533	

Pada Gambar 4.11 sampai 4.13 diberikan grafik yang menunjukkan perubahan banyaknya PTSA tidak isomorfik yang terjadi pada graf kecebong $T_{m,n}$ seiring dengan bertambahnya nilai n . Kemudian pada Gambar 4.14 sampai 4.16 diberikan grafik yang menunjukkan perubahan banyaknya PTSA seiring dengan

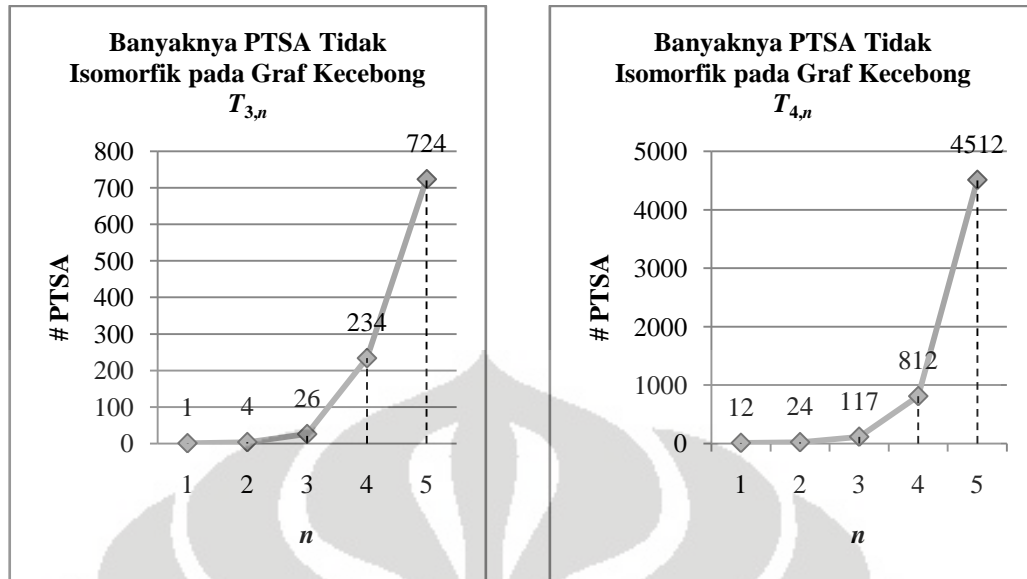
bertambahnya nilai m . Seperti pada grafik banyaknya PTSA untuk graf-graf yang dibahas sebelumnya, terdapat peningkatan yang signifikan untuk banyaknya PTSA tidak isomorfik pada graf kecebong apabila nilai m ataupun nilai n bertambah. Kecenderungan pertambahan banyaknya PTSA ini sesuai dengan grafik fungsi eksponensial sedemikian sehingga untuk nilai m dan n yang lebih besar diperlukan *running time* yang lebih lama. Simulasi untuk nilai m yang lebih besar dari 7 dan nilai n yang lebih besar dari 5 belum dapat dilakukan karena membutuhkan waktu yang cukup lama.

Tabel 4.7 Banyaknya PTSA tidak isomorfik untuk graf $T_{7,n}$, $n = 1$ s.d. 5 dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)}\right)$

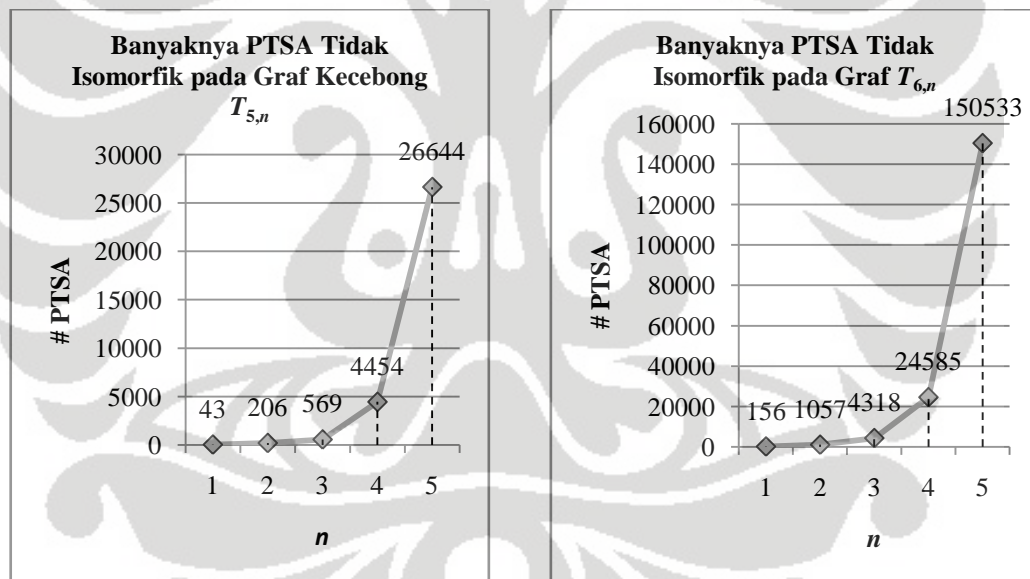
T(7,1)		T(7,2)		T(7,3)		T(7,4)		T(7,5)	
k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA	k	# PTSA
22	31	24	149	27	652	29	1745	32	9643
23	73	25	242	28	976	30	3843	33	26290
24	100	26	578	29	3113	31	9720	34	61296
25	178	27	1285	30	4754	32	17960	35	117873
26	105	28	1119	31	5785	33	28915	36	167509
27	70	29	1311	32	5125	34	32069	37	204423
28	1	30	741	33	3792	35	32559	38	176050
29	0	31	320	34	1355	36	19411	39	140642
		32	49	35	237	37	10164	40	63377
				36	0	38	3386	41	26666
						39	365	42	4246
						40	0	43	0
Total : 558		Total : 5794		Total : 25789		Total : 160137		Total : 998015	

Tabel 4.8 Eksistensi dari PTSA graf $T_{m,n}$ berdasarkan simulasi program dengan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)}\right)$

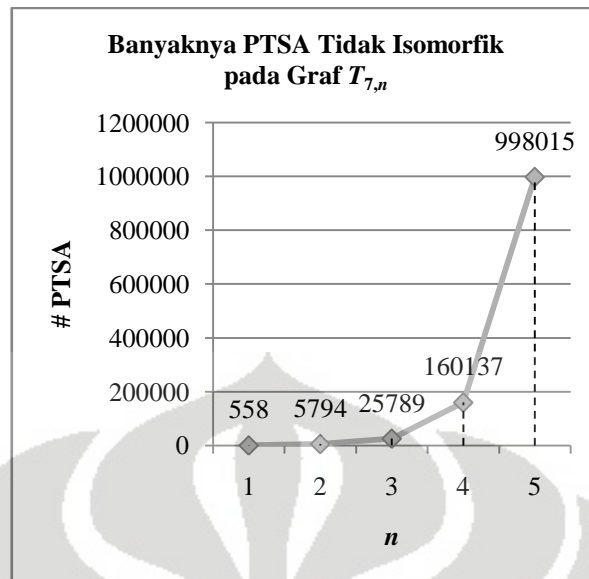
$m \backslash n$	1	2	3	4	5
3	14	14, 16, 17	17 s.d 21	19 s.d. 25	22 s.d 28
4	14 s.d 17	17 s.d 21	19 s.d 24	22 s.d 28	24 s.d 32
5	17 s.d 21	19 s.d 25	22 s.d 28	24 s.d 32	27 s.d 35
6	19 s.d 24	22 s.d 28	24 s.d 32	27 s.d 35	29 s.d 39
7	22 s.d 28	24 s.d 32	27 s.d 35	29 s.d 39	32 s.d 42



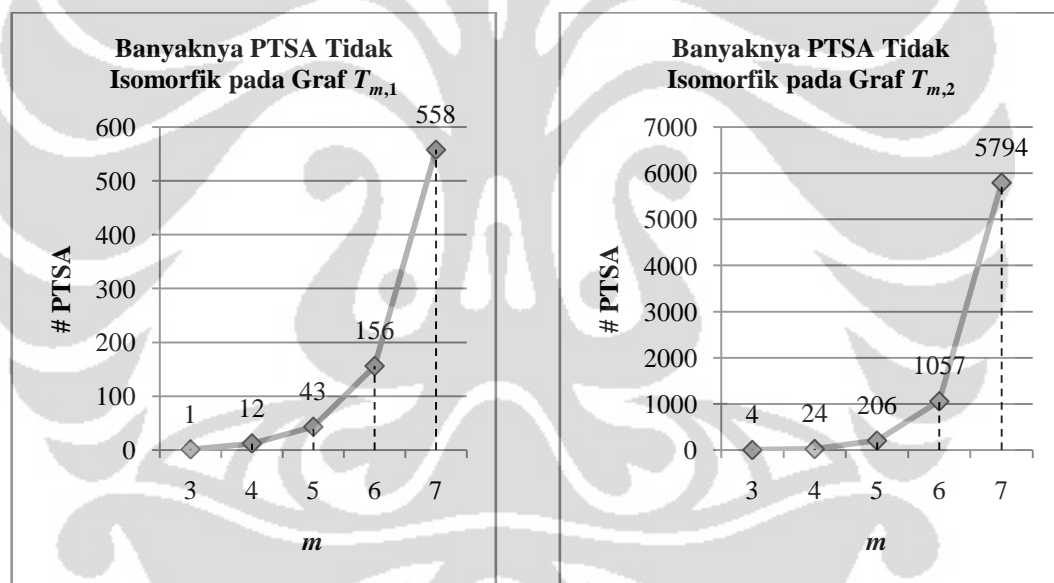
Gambar 4. 11 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{3,n}$ dan $T_{4,n}$ seiring dengan bertambahnya nilai n



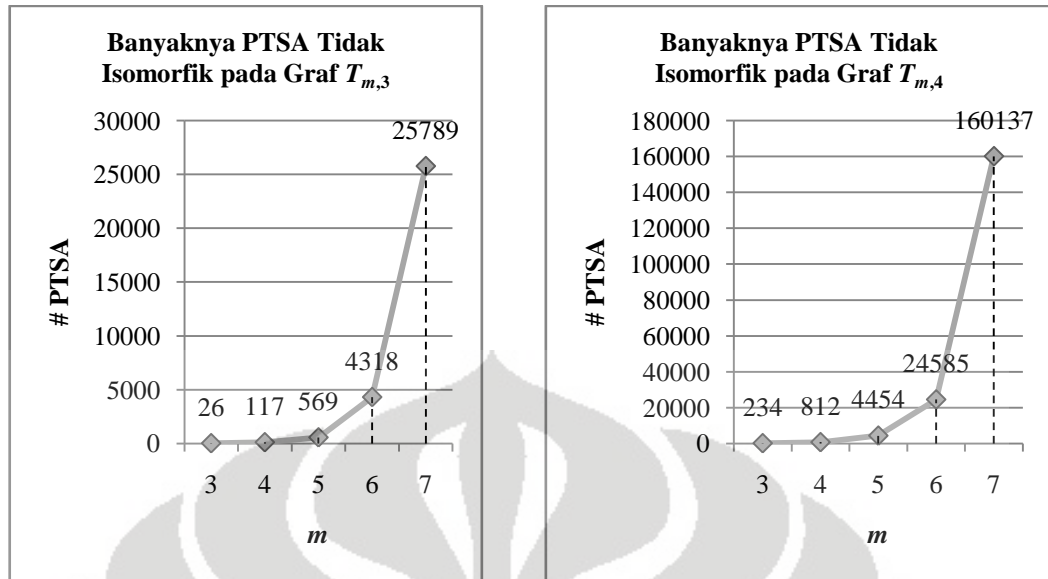
Gambar 4. 12 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{5,n}$ dan $T_{6,n}$ seiring dengan bertambahnya nilai n



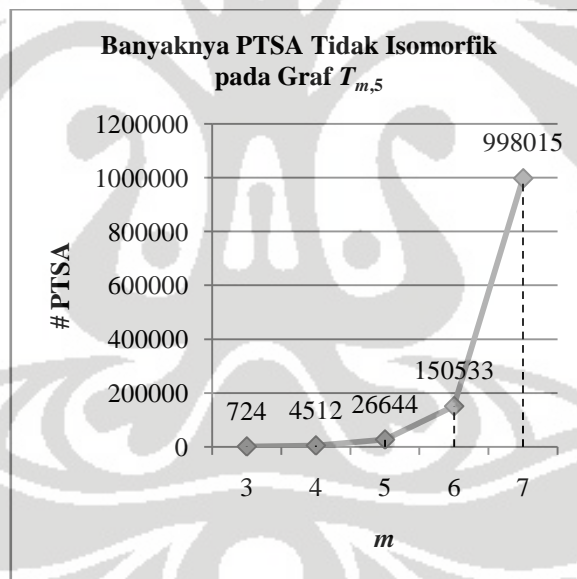
Gambar 4. 13 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{7,n}$ seiring dengan bertambahnya nilai n



Gambar 4. 14 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{m,3}$ dan $T_{m,4}$ seiring dengan bertambahnya nilai m



Gambar 4. 15 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{m,3}$ dan $T_{m,4}$ seiring dengan bertambahnya nilai m



Gambar 4. 16 Grafik perubahan banyaknya PTSA tidak isomorfik pada graf $T_{m,5}$ seiring dengan bertambahnya nilai m

BAB 5 KESIMPULAN

Dalam skripsi ini telah dibangun algoritma-algoritma PTSA untuk graf lingkaran C_n , matahari $C_n \odot \bar{K}_1$, dan kecebong $T_{m,n}$ dengan batasan nilai k masing-masing $\frac{5n+3}{2} \leq k \leq \frac{7n+3}{2}$, $5n+2 \leq k \leq 6n+1$, dan $\max\left(\frac{5(m+n)+3}{2}, 10\right) \leq k \leq \min\left(\frac{7(m+n)+3}{2}, \frac{(m+n)(7(m+n)-3)-14}{2(m+n-1)}\right)$. Algoritma-algoritma ini telah diimplementasikan dalam bentuk program dan sebagai hasil implementasinya, dilakukan simulasi yang memberikan banyaknya PTSA tidak isomorfik untuk semua nilai k yang mungkin dari graf terkait.

Tabel 5.1 Eksistensi PTSA untuk graf lingkaran C_n , matahari $C_n \odot \bar{K}_1$, dan kecebong $T_{m,n}$

Graf	Nilai k yang mungkin	Nilai k PTSA ada	Graf	Nilai k yang mungkin	Nilai k PTSA ada
C_3	9 s.d. 12	9 s.d. 12	$T_{4,2}$	17 s.d. 22	17 s.d. 21
C_4	12 s.d. 15	12 s.d. 15	$T_{4,3}$	19 s.d. 25	19 s.d. 24
C_5	14 s.d. 19	14, 16, 17, 19	$T_{4,4}$	22 s.d. 29	22 s.d. 28
C_6	17 s.d. 22	17 s.d. 22	$T_{4,5}$	24 s.d. 32	24 s.d. 32
C_7	19 s.d. 26	19 s.d. 26	$T_{5,1}$	17 s.d. 22	17 s.d. 21
C_8	22 s.d. 29	22 s.d. 29	$T_{5,2}$	19 s.d. 25	19 s.d. 25
C_9	24 s.d. 33	24 s.d. 33	$T_{5,3}$	22 s.d. 29	22 s.d. 29
C_{10}	27 s.d. 36	27 s.d. 36	$T_{5,4}$	24 s.d. 32	24 s.d. 32
$C_3 \odot \bar{K}_1$	17, 18, 19	18, 19	$T_{5,5}$	27 s.d. 36	27 s.d. 35
$C_4 \odot \bar{K}_1$	22 s.d. 25	23, 24, 25	$T_{6,1}$	19 s.d. 25	19 s.d. 24
$C_5 \odot \bar{K}_1$	27 s.d. 31	28 s.d. 31	$T_{6,2}$	22 s.d. 29	22 s.d. 28
$C_6 \odot \bar{K}_1$	32 s.d. 37	33 s.d. 37	$T_{6,3}$	24 s.d. 32	24 s.d. 32
$C_7 \odot \bar{K}_1$	27 s.d. 43	38 s.d. 43	$T_{6,4}$	27 s.d. 36	27 s.d. 35
$T_{3,1}$	12, 13, 14	14	$T_{6,5}$	29 s.d. 40	29 s.d. 39
$T_{3,2}$	14 s.d. 18	14, 16, 17	$T_{7,1}$	22 s.d. 29	22 s.d. 28
$T_{3,3}$	17 s.d. 22	17 s.d. 21	$T_{7,2}$	24 s.d. 32	24 s.d. 32
$T_{3,4}$	19 s.d. 25	19 s.d. 25	$T_{7,3}$	27 s.d. 36	27 s.d. 35
$T_{3,5}$	22 s.d. 29	22 s.d. 28	$T_{7,4}$	29 s.d. 40	29 s.d. 39
$T_{4,1}$	14 s.d. 18	14 s.d. 17	$T_{7,5}$	32 s.d. 43	32 s.d. 42

Berdasarkan hasil-hasil simulasi dapat disimpulkan bahwa secara umum pertumbuhan banyaknya PTSA yang tidak isomorfik meningkat secara eksponensial seiring dengan bertambahnya nilai n (dan m untuk PTSA graf kecebong). Ha-

sil simulasi juga menunjukkan adanya PTSA dari graf-graf yang dibahas dengan nilai n dan k seperti yang disarikan pada Tabel 5.1. Untuk nilai-nilai k yang tidak memiliki PTSA, perlu mendapatkan penelitian lebih lanjut.



DAFTAR PUSTAKA

- Baker, A., & Sawada, J. (2008). Magic labelings on cycles and wheels. *COCOA LNCS* , 361-373.
- Gallian, J. A. (2009). A dynamic survey of graph labeling. *The Electronic Journal of Combinatorics* 5 , #DS6.
- Graph Isomorphism*. (n.d.). Retrieved Februari 22, 2010, from <http://www.cs.uu.nl/docs/vakken/an/>
- MacDougall, J. A., Miller, M., & Wallis, W. D. (2002). Vertex-magic of wheels and related graphs. *Util. Math.* 62 , 175-183.
- MacDougall, J. A., Miller, M., Slamin, & Wallis, W. D. (2002). Vertex-magic total labelings of graphs. *Util. Math.* 61 , 3-21.
- Rahim, M. T., & Slamin. (2008). Vertex-magic total labeling of the union of sun.
- Rosen, K. H. (1995). *Discrete mathematics and its applications* (3 ed.). New York: McGraw-Hill.
- Slamin, Prihandoko, A. C., Setiawan, T. B., Rosita, F., & Shaleh, B. (2006). Vertex-magic total labeling of disconnected graphs. *Journal of Prime Research in Mathematics Vol 2* , 147-156.

Lampiran 1 Tampilan masukan dan keluaran untuk PTSA C_6

• Tampilan Masukan

```

Command Window
>> cycle

      Program Pelabelan Total Simpul Ajaib (PTSA) Graf Lingkaran
      ~~~~~
Program ini bertujuan untuk mengetahui banyaknya PTSA Lingkaran(n,k)
yang tidak isomorfik

Masukan banyaknya simpul pada lingkaran (n): 6
Batasan konstanta ajaib k adalah [17,22]
Masukan konstanta ajaib k: 17
Masukan nama berkas keluaran: C6.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
1

Masukan banyaknya simpul pada lingkaran (n): 6
Batasan konstanta ajaib k adalah [17,22]
Masukan bobot simpul (k): 18
Masukkan nama berkas keluaran: C6.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
1

Masukan banyaknya simpul pada lingkaran (n): 6
Batasan konstanta ajaib k adalah [17,22]
Masukan bobot simpul (k): 19
Masukkan nama berkas keluaran: C6.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
1

Masukan banyaknya simpul pada lingkaran (n): 6
Batasan konstanta ajaib k adalah [17,22]
Masukan bobot simpul (k): 20
Masukkan nama berkas keluaran: C6.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
1

Masukan banyaknya simpul pada lingkaran (n): 6
Batasan konstanta ajaib k adalah [17,22]
Masukan bobot simpul (k): 21
Masukkan nama berkas keluaran: C6.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
1

Masukan banyaknya simpul pada lingkaran (n): 6
Batasan konstanta ajaib k adalah [17,22]
Masukan bobot simpul (k): 22
Masukkan nama berkas keluaran: C6.doc

Ingin mencoba lagi?
Tekan 1 jika ingin mencoba lagi.
Tekan 2 jika tidak.
2
Bye...
A >>

```

• Tampilan Keluaran

```

Screen 1 of 6
document, open the latest version or save a copy. Save a Copy

1.
Label simpul:
4 8 12 10 11 9
Label busur:
6 3 2 5 1 7

2.
Label simpul:
4 10 11 9 12 8
Label busur:
6 1 5 3 2 7

3.
Label simpul:
6 12 10 8 11 7
Label busur:
2 3 4 5 1 9

Banyaknya PTSA Lingkaran(6,17) = 3
=====

1.
Label simpul:
3 11 12 6 9 7
Label busur:

Screen 2 of 6
document, open the latest version or save a copy. Save a Copy

5 2 4 8 1 10

Banyaknya PTSA Lingkaran(6,18) = 1
=====

1.
Label simpul:
1 9 12 10 8 2
Label busur:
7 3 4 5 6 11

2.
Label simpul:
1 10 8 6 12 5
Label busur:
7 2 9 4 3 11

3.
Label simpul:
2 6 10 11 9 4
Label busur:
5 8 1 7 3 12

4.
Label simpul:
2 11 9 4 10 6

```

(lanjutan)

```

Screen 3 of 6
document, open the latest version or save a copy. Save a Copy

Label busur:
5 3 7 8 1 12

5.
Label simpul:
2 12 10 4 9 5
Label busur:
6 1 8 7 3 11

6.
Label simpul:
2 10 7 5 12 6
Label busur:
8 1 11 3 4 9

Banyaknya PTSA Lingkaran(6,19) = 6
=====

1.
Label simpul:
1 11 8 4 9 3
Label busur:
7 2 10 6 5 12

2.
Label simpul:

```

```

Screen 4 of 6
document, open the latest version or save a copy. Save a Copy

1 3 5 11 12 4
Label busur:
9 8 7 2 6 10

3.
Label simpul:
1 7 5 3 12 8
Label busur:
9 4 11 6 2 10

4.
Label simpul:
1 7 11 3 6 8
Label busur:
9 4 5 12 2 10

5.
Label simpul:
2 4 9 11 7 3
Label busur:
6 10 1 8 5 12

6.
Label simpul:
2 11 7 3 9 4
Label busur:
8 1 12 5 6 10

```

```

Screen 5 of 6
document, open the latest version or save a copy. Save a Copy

Banyaknya PTSA Lingkaran(6,20) = 6
=====

1.
Label simpul:
1 7 4 6 10 2
Label busur:
9 5 12 3 8 11

Banyaknya PTSA Lingkaran(6,21) = 1
=====

1.
Label simpul:
1 3 5 2 6 7
Label busur:
10 9 8 12 4 11

2.
Label simpul:
1 4 2 3 9 5
Label busur:
10 8 12 7 6 11

```

```

Screen 6 of 6
document, open the latest version or save a copy. Save a Copy

3.
Label simpul:
1 5 9 4 2 3
Label busur:
10 7 6 12 8 11

Banyaknya PTSA Lingkaran(6,22) = 3
=====

```