DIMENSI PARTISI GRAF HASIL KALI 2-KUAT DAN PROGRAM PENCARI PARTISI PEMBEDA BEBERAPA KELAS GRAF

TUGAS AKHIR

Karya tulis sebagai salah satu syarat

untuk memperoleh gelar Sarjana

dari Institut Teknologi Bandung

Oleh

ILMA ALIYA FIDDIEN

NIM: 10117019

(Program Studi Sarjana Matematika)

INSTITUT TEKNOLOGI BANDUNG

Juni 2021

ABSTRAK

DIMENSI PARTISI GRAF HASIL KALI 2-KUAT DAN PROGRAM PENCARI PARTISI PEMBEDA BEBERAPA KELAS GRAF

Oleh

ILMA ALIYA FIDDIEN

NIM: 10117019

(Program Studi Matematika)

Misalkan G adalah suatu graf non trivial dengan himpunan titik $V(G)$ dan himpunan sisi $E(G)$. Untuk tiap $v \in V(G)$ dan himpunan $S \subset V(G)$, jarak antara v dan S, $d(v,S)$ adalah jarak terpendek antara v dan titik-titik anggota S. Representasi v terhadap partisi titik $\Pi = \{S_1, S_2, ..., S_k\}$ adalah k-vektor $r(s|\Pi) = (d(s,S_1), d(v,S_2), ..., d(v,S_k))$. Π adalah partisi pembeda dari G jika semua representasi setiap titik di G terhadap Π berbeda. Dimensi partisi G, $pd(G)$, adalah kardinalitas terkecil dari partisi pembeda yang mungkin untuk G. Pada penelitian ini, untuk G dan H dua sebarang graf berdiameter minimal k, didefinisikan graf hasil kali k-kuat $G \boxtimes_k H$, suatu perumuman dari graf hasil kali kuat. Dengan mengkaji pola diameternya, diberikan rumus diameter untuk graf hasil kali 2-kuat dari graf lintasan, lingkaran, dan bipartite lengkap. Kemudian ditentukan batas-batas dimensi partisi graf hasil kali 2-kuat menggunakan keterangan orde dan diameter graf-graf asalnya. Graf kuasa ke-2 juga dilibatkan untuk membantu memberikan batas atas bagi $G \boxtimes_2 H$. Di bagian akhir, ditunjukkan program pencari himpunan pembeda untuk beberapa kelas graf yang dibuat menggunakan bahasa pemrograman Python dan digunakan selama penelitian ini.

Kata Kunci: partisi pembeda, dimensi partisi, graf hasil kali k-kuat, graf kuasa

ABSTRACT

PARTITION DIMENSION ON 2-STRONG PRODUCT GRAPHS AND RESOLVING PARTITION FINDER PROGRAM FOR SOME CLASSES OF GRAPH

By

ILMA ALIYA FIDDIEN

NIM: 10117019

(Undgraduate Program in Mathematics)

Let G be a non-trivial graph with vertices set $V(G)$ and edges set $E(G)$. For each $v \in V(G)$ and $S \subset V(G)$, distance between v and S, $d(v, S)$ is the shortest distance between v and every vertices in S. Representation of v with respect to an ordered partition $\Pi = \{S_1, S_2, ..., S_k\}$ is k-vector $r(s|\Pi) = (d(s, S_1), d(v, S_2), ..., d(v, S_k))$. Π is resolving partition for G if all representation of vertices of G with respect to Π is unique. Dimension partition of G, or $pd(G)$, is the smallest cardinality of resolving partitions for G. In this research, for G and H any graph with minimum diameter k, we define k-strong product graph, $G \boxtimes_k H$, a generalization of strong product graph. Through pattern-finding, we formulate the diameter of 2-strong product graphs of path, cycle, and complete bipartite graph. We then determine boundaries for partition dimension of 2-strong product graphs, utilizing the order and diameter of the original graphs. We involved 2nd power graph to help find resolving partition for $G \boxtimes_2 H$. In the last section, we show a resolving partition finder program for few graph classes which written in Python and were used to help the research.

Keywords: resolving partition, partition dimension, k-strong product graph, power graph

LEMBAR PENGESAHAN

DIMENSI PARTISI GRAF HASIL KALI 2-KUAT DAN PROGRAM PENCARI PARTISI PEMBEDA BEBERAPA KELAS GRAF

Laporan Tugas Akhir

Oleh

Ilma Aliya Fiddien

10117019

(Program Studi Sarjana Matematika)

Telah disetujui dan disahkan sebagai Laporan Tugas Akhir
di Bandung pada tanggal ... Juni 2020.

Pembimbing

Prof. Dr. M. Salman A.N., S.Si., M.Si.
NIP 19680916 199402 1 001

Dengan menyebut nama Allah Yang Maha Pengasih dan Maha Penyayang.
Dipersembahkan untuk orang tua dan guru-guruku, semoga Allah limpahkan pahala dari tugas akhir ini kepada mereka.

KATA PENGANTAR

Assalamu'alaikum warahmatullahi wabarakatuh.

Alhamdulillaahil-ladzii bini'matihi tatimmush-saalihaat. Segala puji dan syukur tidak akan pernah cukup terbilang untuk Sang Maha Memiliki Ilmu yang telah memberikan petunjuk dan kemudahan kepada penulis untuk menyelesaikan Tugas Akhir ini. Banyak sekali pihak yang Allah lembutkan hati dan mudahkan tangannya untuk mendukung penulis dalam menuntaskan amanahnya sebagai mahasiswa sarjana matematika, sejak awal hingga akhir. Izinkanlah penulis mengucapkan beberapa ucapan dan doa kepada mereka, sebagai bentuk rasa hormat atas jasa mereka.

1. Ibunda, Waryamah, dan ayahanda, Sahmudin, tidak pernah berhenti memberikan perhatian dan dukungan kepada penulis di sepanjang hayatnya. Niat dan usaha mereka mengirimkan Penulis berkuliah di ITB agar menjadi anak yang lebih shalehah dan semakin bermanfaat bagi sesamanya adalah motivasi terbesar Penulis untuk tetap bertahan menjalani empat tahun perkuliahan. Semoga Allah kumpulkan lagi kami dan sekeluarga di surga-Nya yang kekal. *Rabbighfirli waliwalidayya warhamhuma kama rabbayani shaghiran.*
2. Prof. M. Salman A. N., M.Si, selaku dosen pembimbing tugas akhir Penulis sekaligus dosen matematika pertama bagi Penulis. Sejak tahap persiapan pertama, beliau telah menginspirasi Penulis untuk terus menikmati proses belajar, menjadi manusia yang berkarakter, dan terus berbuat baik dan mendoakan kedua orang tua. Semoga Pak Salman dan keluarga selalu dalam keadaan sehat dan selamat.
3. Seluruh dosen program studi Matematika ITB dan dosen-dosen yang telah mengajari Penulis di Institut Teknologi Bandung. Semoga ilmu yang mereka ajarkan dapat kembali menjadi pahala jariyah.
4. Keluarga Asrama Salman ITB 2018/2019 dan 2019/2020, telah bertukar banyak motivasi dan pelajaran hidup yang membantu membuat Penulis menjadi individu yang lebih berempati pada lingkungan sekitar. Semoga setiap

dari mereka menjadi orang-orang yang semakin menginspirasi sekitarnya.

5. Teman-teman Penulis di Kepengurusan GAMAIS ITB 2021, 2020, 2018, dan 2017 yang secara langsung mau tidak langsung memotivasi Penulis untuk tetap dekat dengan Sang Pencipta dan berlatih menjadi muslim yang profesional.
6. Teman-teman Penulis di P3RI Salman ITB 1440H dan 1439H, tim IC, tim formatur, dan tim sekretaris, para pengurus dan pembina YPM Salman ITB, dan BMKA Salman, serta Beasiswa Aktivis Salman yang telah menyajikan pengalaman hidup bagi Penulis yang masih banyak belajar menjadi manusia yang tulus berkontribusi melayani masyarakat serta tidak takut untuk bermimpi besar.

Penulis menyadari bahwa masih banyak hal yang bisa diperbaiki dan dikembangkan dari tugas akhir ini. Oleh karena itu, kritik dan saran dari pembaca akan sangat berharga bagi Penulis, baik dari segi penulisan maupun penelitian. Penulis berharap buku tugas akhir ini bisa bermanfaat kepada pembaca dalam bentuk apapun yang mungkin tidak akan pernah Penulis duga.

Wassalamu'alaikum warahmatullahi wabarakatuh.

Majalengka, 9 Juni 2021
Penulis

Ilma Aliya Fiddien

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR PUSTAKA

DAFTAR LAMBANG

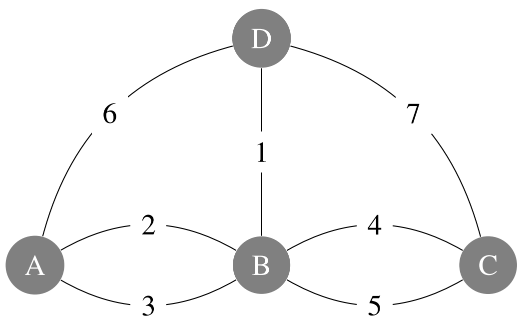
$G(V, E)$	Graf G dengan himpunan titik V dan himpunan sisi E
$V(G)$	Himpunan titik dari graf G
$E(G)$	Himpunan sisi dari graf G
$e = uv$	Sisi e yang menghubungkan titik u dan titik v
$\|G\|$	Orde (banyak titik) dari graf G
$\|\|G\|\|$	*Size* (banyak sisi) dari graf G
δ	Derajat terkecil suatu graf
Δ	Derajat terbesar suatu graf
$d(u, v)$	Jarak antara titik u dan titik v
$d(u, \Pi)$	Jarak antara titik u dan himpunan/partisi himpunan titik Π
$r(u\|\Pi)$	Representasi titik u terhadap himpunan/partisi himpunan titik Π
$diam(G)$	Diameter dari graf G
$dim(G)$	Dimensi metrik dari graf G
$pd(G)$	Dimensi partisi dari graf G
P_n	Graf lintasan orde n
C_n	Graf lingkaran orde n
K_n	Graf lengkap orde n
$K_{m,n}$	Graf bipartit lengkap orde $m + n$
$\square$	Operator kali kartesius
$\boxtimes$	Operator kali kuat
$\boxtimes_n$	Operator kali n-kuat

Bab 1 Pendahuluan

1.1 Latar Belakang

Banyak permasalah dapat direpresentasikan oleh sehimpun *titik* beserta sehimpun *sisi* yang menghubungkan titik-titik tersebut. Contohnya, titik merepresentasikan individu di suatu komunitas dan sisi merepresentasikan hubungan pertemanan antar individu. Dalam contoh ini, kita dapat mencari tahu apakah sepasang titik terhubung oleh suatu sisi, dengan kata lain, apakah dua individu tersebut berteman atau tidak. Objek yang direpresentasikan oleh titik dan garis ini dapat bersifat konkrit maupun abstrak. Abstraksi inilah yang melahirkan teori graf.

Konsep graf diperkenalkan oleh Leonhard Euler pada tahun 1736 dengan masalah Jembatan Königsberg. Euler memisalkan empat daratan di sekitar Sungai Pregel sebagai titik dan ketujuh jembatannya sebagai sisi. Dia meneliti apakah mungkin untuknya membuat suatu perjalanan atau *tour* yang dapat melewati seluruh jembatan cukup sekali sekali saja. *Tour* demikian disebut *Euler tour*. Telah terbukti bahwa *tour* tersebut tidak mungkin ada untuk kasus ini. Masalah ini dimodelkan sebagai graf berikut.



Gambar 1.1: Graf Jembatan Königsberg

Graf G adalah sebagai suatu pasangan terurut $(V(G), E(G))$. Himpunan titik $V(G)$ terdiri dari titik-titik unik tak terurut pada G. Himpunan sisi $E(G)$ terdiri atas sisi-sisi unik tak terurut pada G yang menghubungkan titik-titik pada $V(G)$. Pilih G sebagai graf seperti yang terlihat pada Gambar 1.1, maka $V(G) = \{A, B, C, D\}$

dan $E(G) = \{1, 2, 3, 4, 5, 6, 7\}$.

Dalam banyak aplikasi, graf juga sering digunakan untuk memodelkan jaringan atau *network*. Dalam mensistemasikan jaringan, diperlukan cara untuk mendeteksi lokasi suatu benda pada jaringan tersebut. Contohnya, robot bernavigasi dari titik ke titik pada suatu jaringan. Namun dalam graf atau jaringan tidak ada konsep arah yang universal seperti pada bidang Euclid. Misalkan robot berkomunikasi dengan sejumlah sensor-sensor yang diletakkan di titik-titik pada jaringan. Titik-titik yang ditempati sensor tersebut kita sebut *landmarks*. Sensor-sensor tersebut akan memberikan keterangan jarak robot terhadap *landmarks* untuk memfasilitasi navigasi robot. Dengan cara ini, tujuan yang ingin dicapai adalah bagaimana cara memilih titik-titik sebagai *landmarks* sehingga setiap titik di jaringan tersebut dapat ditentukan lokasinya secara unik oleh *landmarks* tersebut. Lebih jauhnya lagi, jika diberikan suatu jaringan, kita ingin mengetahui berapa banyak minimal sensor yang dibutuhkan sehingga setiap titik punya alamat unik. Inilah motivasi dimunculkannya konsep dimensi metrik yang dipublikasikan oleh Slater (1975) dan Harary & Melter (1976) secara independen.

Mencari dimensi metrik suatu graf adalah mencari sub himpunan titik berkardinalitas terkecil dari graf tersebut sedemikian hingga setiap titik memiliki alamat unik terhadap sub himpunan itu. Himpunan yang membedakan seluruh titik disebut himpunan pembeda. Jika kardinalitasnya terkecil, maka ia disebut himpunan pembeda minimal, dengan kardinalitasnya disebut dimensi metrik graf. Kemudian, Chartrand et al. (2000) mengembangkan konsep dimensi partisi (*partition dimension*) yang mirip dengan dimensi metrik. Mencari dimensi partisi adalah mencari partisi titik berkardinalitas terkecil suatu graf sedemikian sehingga setiap titik di graf tersebut memiliki alamat unik terhadap partisi tersebut. Partisi yang mampu membedakan setiap titik ini disebut partisi pembeda. Jika kardinalitasnya terkecil, maka disebut partisi pembeda minimal, dengan kardinalitasnya disebut dimensi metrik graf. Nilai dimensi partisi suatu graf dapat lebih kecil dari pada dimensi metriknya.

Mencari dimensi metrik graf pohon dapat diselesaikan dalam waktu linear, seperti yang dijelaskan Slater (1975). Namun, Garey (1979) telah membuktikan bahwa masalah pencarian dimensi partisi sebarang graf merupakan masalah NP-Complete. Khuller et al. (1996) menunjukkan bahwa mencari dimensi metrik graf dengan orde n dapat diaproksimasi dengan faktor $O(\log n)$ dalam waktu polinomial. Hal ini mengimplikasikan bahwa pencarian himpunan (dan partisi) pembeda yang minimal hanya bisa dilakukan menggunakan algoritma heuristik atau dengan mencari nilai aproksimasinya untuk kelas-kelas graf tertentu.

Meski menemukan nilai dimensi metrik atau dimensi partisi suatu graf sulit secara komputasional. Namun, mengidentifikasi setiap titik secara unik adalah kemampuan yang sangat berguna. Selain aplikasi pada navigasi robot jarak jauh (Khuller et al., 1996), himpunan pembeda memiliki peran penting pada pemrosesan gambar digital (Melter & Tomescu, 1984), juga digunakan untuk merepresentasikan struktur senyawa kimia pada desain obat (Johnson, 1993), dan sebagai alat untuk menemukan sumber difusi pada suatu jaringan (Spinelli et al., 2017). Selain itu, permainan Mastermind dapat dianalisis melalui dimensi metrik graf Hamming (Chvátal, 1983).

Kajian teoritis dimensi partisi di antaranya adalah karakterisasi graf berorde n dengan dimensi partisi 2, n, n-1 (Chartrand et al., 2000), n-2 (Tomescu, 2008), dan dimensi partisi graf hasil kali kartesius & hasil kali kuat (Yero et al., 2014).

Pada karya tulis ini, akan dicari dimensi partisi graf hasil kali 2-kuat. Untuk suatu k bilangan bulat positif, operasi kali k-kuat dikenalkan sebagai perumuman dari operasi kali kuat. Dipelajari juga graf kuasa ke-2, yang analisis strukturnya berguna untuk mendapatkan batas atas dimensi partisi graf hasil kali 2-kuat. Selain itu, karena penelitian pada graf menuntut representasi visual yang banyak dan pengecekan keunikan partisi pembeda membutuhkan banyak komputasi, akan dibuat program yang membantu penelitian ini.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya, masalah yang dibahas dalam penelitian ini adalah sebagai berikut.

1. Apa batas atas dan bawah dimensi partisi graf hasil kali 2-kuat dua graf sembarang?
2. Berapakah dimensi partisi graf hasil kali 2-kuat dua kelas graf tertentu?
3. Bagaimana cara membuat program yang dapat membantu mencari partisi pembeda beberapa kelas graf tertentu?

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut.

1. Mendapatkan batas atas dan bawah dimensi partisi graf hasil kali 2-kuat dua graf sembarang
2. Menentukan dimensi partisi graf hasil kali 2-kuat dua kelas graf tertentu
3. Membuat program yang dapat membantu mencari partisi pembeda beberapa kelas graf tertentu

1.4 Batasan Masalah

Agar fokus bahasan pada penelitian ini terjaga, masalah dibatasi pada:

1. Graf yang menjadi objek penelitian adalah graf sederhana, terhubung, dan terbatas.
2. Operasi graf yang dikaji adalah kali 2-kuat.
3. Kelas-kelas graf yang dioperasikan kali 2-kuat yang dikaji adalah lintasan, siklus, dan graf bipartit lengkap.
4. Program yang dibuat digunakan untuk membantu visualisasi beberapa kelas graf di atas dan graf hasil kali k-kuat dan kali kartesius

1.5 Metode Penelitian

Untuk menjawab rumusan masalah pertama, diambil langkah-langkah sebagai berikut.

1. Mendefinisikan operasi kali k-kuat graf
2. Mengeksplorasi karakteristik graf hasil kali 2-kuat dua sembarang graf
3. Merumuskan batas atas dan batas bawah dari dimensi partisi graf hasil kali 2-kuat dua sembarang graf
4. Mengecek dan memperbaiki batas atas dan batas bawah yang diperoleh
5. Menyimpulkan hasil penelitian

Untuk menjawab rumusan masalah kedua, diambil langkah-langkah sebagai berikut.

1. Mengeksplorasi karakteristik graf hasil kali 2-kuat beberapa kelas graf
2. Merumuskan batas atas, batas bawah, atau nilai eksak dari dimensi partisi graf hasil kali 2-kuat beberapa kelas graf
3. Menyimpulkan hasil penelitian

Untuk menjawab rumusan masalah ketiga, diambil langkah-langkah sebagai berikut.

1. Menentukan arsitektur program
2. Membuat algoritma program
3. Membuat antarmuka program
4. Mengevaluasi algoritma dan antarmuka program dan memperbaikinya

1.6 Sistematika Pembahasan

Sistematika penulisan karya tulis tugas akhir ini adalah sebagai berikut.

Bab pertama, yaitu Bab Pendahuluan, terdiri dari sub bab penjelasan latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, metode penelitian, dan sistematika pembahasan.

Bab kedua, yakni Bab Landasan Teori, membahas dasar teori yang digunakan pada penelitian, yaitu definisi graf, jenis graf, beberapa kelas graf, beberapa operasi pada graf, definisi dimensi partisi, serta hasil penelitian terdahulu yang berkaitan dengan dimensi partisi yang mendukung penelitian ini.

Bab ketiga, yaitu Bab Dimensi Partisi Graf Hasil Kali 2-Kuat, membahas definisi graf hasil kali k-kuat, hubungan graf hasil kali 2-kuat dengan graf hasil kali lainnya dan graf kuasa ke-2 dari suatu graf, dan hasil penelitian.

Bab keempat, yaitu Bab Program Pencari Partisi Pembeda Graf-Graf Tertentu, membahas perancangan program, arsitektur program, antarmuka program, dan fitur program pencari partisi pembeda graf-graf tertentu.

Bab kelima, yaitu Bab Kesimpulan dan Saran, menyimpulkan hasil yang dapat menjawab pertanyaan pada bagian rumusan masalah. Selain itu, diberikan pula saran untuk penelitian selanjutnya mengenai graf hasil kali 2-kuat dan dimensi partisinya.

Bab 2 Landasan Teori

2.1 Terminologi pada Teori Graf

2.1.1 Definisi Graf

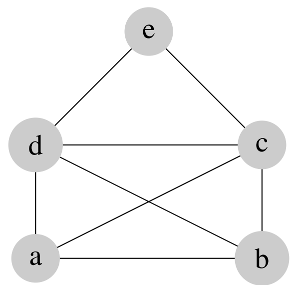
Suatu **graf** G adalah sistem pasangan terurut $(V(G), E(G))$. $V(G)$ adalah himpunan *titik* (*vertex*) dan $E(G)$ adalah himpunan *sisi* (*edge*) dengan $V(G) \cap E(G) = \emptyset$. *Fungsi insidensi* dari G, $\psi(G)$, didefinisikan sebagai pemetaan $E(G)$ ke $(V(G))^2$. Titik $u, v \in V(G)$ dikatakan *bertetangga* dan $e \in E(G)$ dikatakan *menghubungkan* u dan v jika memenuhi $\psi(e) = \{u, v\}$, dengan titik u dan v disebut ujung dari sisi e. Banyaknya titik (*orde*) dan banyaknya sisi (*size*) dari graf G masing-masing dinotasikan $|V(G)|$ dan $|E(G)|$. Himpunan titik yang bertetangga dengan titik v ditulis sebagai $N(v)$. Banyaknya sisi yang bertetangga dengan titik v disebut dengan *derajat* dari v atau $deg(v)$. Suatu graf dikatakan berarah (*directed*) jika sisi $\{u, v\}$ dipandang sebagai himpunan terurut, yang berarti dari sisi e menghubungkan dari u ke v. Sebaliknya, pada graf tak berarah (*undirected*), sisi $\{u, v\} = \{v, u\}$.

Untuk kemudahan penulisan, notasi $\psi(e) = \{u, v\}$ akan sering ditulis sebagai $e = uv$.

Contoh 1. Graf amplop adalah graf yang didefinisikan sebagai berikut.

$$V(H) = \{a, b, c, d, e\}, \qquad E(H) = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\},$$

dengan $e_1 = ab, e_2 = bc, e_3 = cd, e_4 = ad, e_5 = de, e_6 = ce, e_7 = ac, e_8 = bd$.



Gambar 2.1: Graf amplop

Suatu graf dikatakan **sederhana** jika tidak mengandung sisi ganda (*multiple edges*) dan sisi gelang (*loop*). Graf amplop di atas adalah graf sederhana.

Sebuah **lintasan** dengan panjang n pada suatu graf G adalah barisan sisi-sisi terurut $v_i v_{i+1} = e_i \in E(G)(i = 0, 1...n-1)$, dan dituliskan sebagai $v_0 v_1 ... v_{n-1} v_n$.

Sebuah graf G dikatakan **terhubung** (*connected*) jika untuk setiap pasang titik di G terdapat setidaknya satu lintasan yang menghubungkan kedua titik tersebut.

2.1.2 Graf Isomorfis

Graf dapat direpresentasikan melalui ilustrasi dengan geometri yang berbeda-beda. Graf G_1 dan G_2 dikatakan isomorfis jika terdapat pemetaan satu-satu $g : V(G_1) \rightarrow V(G_2)$ sedemikian sehingga $g(u)$ dan $g(v)$ bertetangga di G_2 jika dan hanya jika u dan v bertetangga di G_1.

2.1.3 Jarak pada Graf

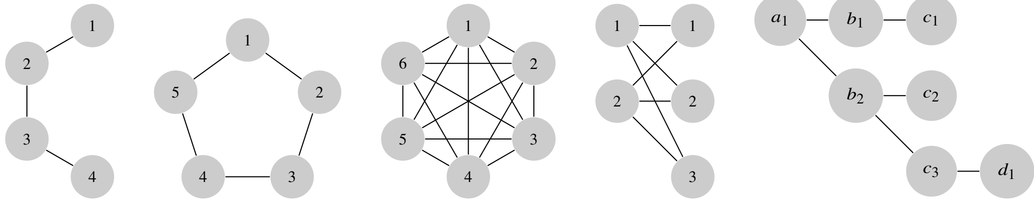
Jarak antara dua titik $u, v \in V(G)$, ditulis $d_G(u, v))$ adalah panjang lintasan terpendek di graf G yang menghubungkan u dan v. Sedangkan jarak antara titik $u \in V(G)$ dan himpunan $S \subseteq V(G)$ adalah $d_G(u, S)) = \underset{v \in V(G)}{min} \{d(u, v)\}$. Jika lintasan tersebut ada, maka $d_G(u, v)) < \infty$, dan $d_G(u, v)) = \infty$ jika sebaliknya. Untuk konteks graf yang jelas, jarak antara u dan v di graf G cukup ditulis $d(u, v)$.

Eksentrisitas $\epsilon(v)$ suatu titik v pada graf G adalah jarak terbesar v dengan titik-titik lainnya di G, atau $\epsilon(v) = \underset{u \in V(G)}{max} \{d(v, u)\}$. Radius dari suatu graf G adalah eksentrisitas minimum dari titik-titiknya, atau $rad(G) = \underset{v \in V(G)}{min} \{\epsilon(v)\}$. Sedangkan **diameter** dari graf G adalah eksentrisitas maksimum dari titik-titiknya, atau $diam(G) = \underset{v \in V(G)}{max} \{\epsilon(v)\}$.

2.2 Jenis-Jenis Graf

2.2.1 Graf Lintasan

Graf lintasan (*path*) berorde n, ditulis P_n, adalah graf terhubung yang hanya terdiri dari lintasan dengan panjang n. Dengan kata lain, setiap titik-titiknya berderajat dua, kecuali dua titik ujung yang berderajat satu. Diameter dari P_n adalah n.



Gambar 2.2: Dari kiri ke kanan: $P_4, C_5, K_6, K_{2,3}$, graf pohon

2.2.2 Graf Lingkaran

Graf lingkaran (*cycle*) C_n adalah graf terhubung berorde n dengan seluruh titiknya berderajat dua. Diameter dari C_n adalah $\lceil n/2 \rceil$.

2.2.3 Graf Lengkap

Graf lengkap (*complete*) K_n adalah graf terhubung berorde n dengan seluruh titiknya saling bertetangga. Diameter dari graf lengkap selalu 1.

2.2.4 Graf Bipartit Lengkap

Graf multipartit (*multipartite*) adalah graf terhubung yang himpunan titiknya dibagi menjadi beberapa partisi sedemikian sehingga titik-titiknya hanya bertetangga dengan partisi selain partisinya sendiri. Jika partisinya sejumlah n maka diameternya adalah n. Jika partisi titiknya hanya dua maka disebut graf bipartit.

Graf bipartit lengkap atau $K_{n,m}$ adalah graf yang setiap titik di suatu partisi bertetangga dengan setiap titik di partisi lainnya. Graf bintang (*star*) S_n adalah graf bipartit lengkap $K_{1,n}$ dengan orde $n + 1$.

2.3 Operasi pada Graf

Sebagai objek matematis pada umumnya, graf juga dapat dioperasikan secara uniter maupun biner. Operasi uniter hanya membutuhkan satu graf untuk menghasilkan suatu graf baru, sedangkan operasi biner memerlukan dua graf untuk menghasilkan suatu graf baru. Karya tulis ini akan berfokus mendefinisikan operasi *kali k-kuat*. Sebelum itu, perlu dipahami terlebih dahulu definisi dari graf hasil *kali Kartesius* dan hasil *kali kuat* dari sepasang graf dan graf *kuasa ke-k* dari suatu graf.

2.3.1 Operasi Kali Kartesius

Mengacu pada Puš (1991), graf hasil kali kartesius dan graf hasil kali kuat didefinisikan sebagai berikut.

Definisi 1. Untuk graf G dan H, **graf hasil kali Kartesius** $G \square H$ adalah graf yang memiliki himpunan titik $V(G) \times V(H)$, dengan titik (u,x) dan (v,y) bertetangga jika dan hanya jika

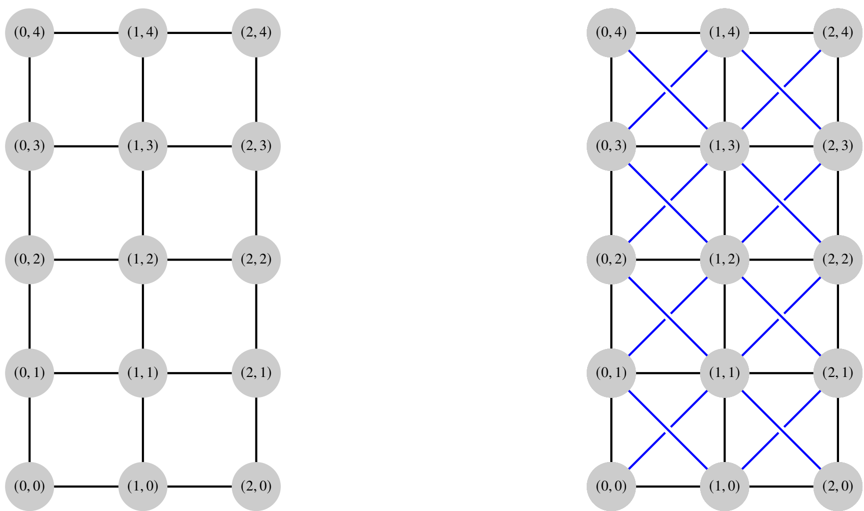
- $uv \in E(G)$ dan $x = y$, atau
- $u = v$ dan $xy \in E(H)$.

2.3.2 Operasi Kali Kuat

Definisi 2. Untuk graf G dan H, **graf hasil kali kuat** $G \boxtimes H$ adalah graf yang memiliki himpunan titik $V(G) \times V(H)$, dengan titik (u,x) dan (v,y) bertetangga jika dan hanya jika

- $uv \in E(G)$ dan $x = y$, atau
- $u = v$ dan $xy \in E(H)$, atau
- $uv \in E(G)$ dan $xy \in E(H)$.

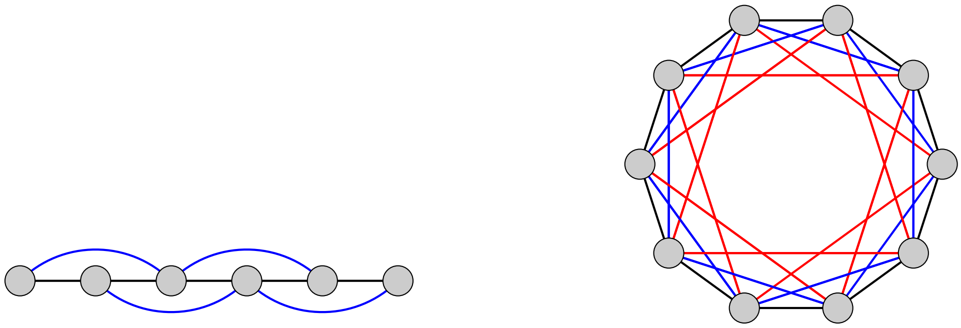
Pilihan syarat terakhir dapat dituliskan ulang menggunakan konsep jarak antara dua titik menjadi $d_G(u,v) = 1$ dan $d_H(x,y) = 1$.



(a) Graf $P_5 \square P_3$

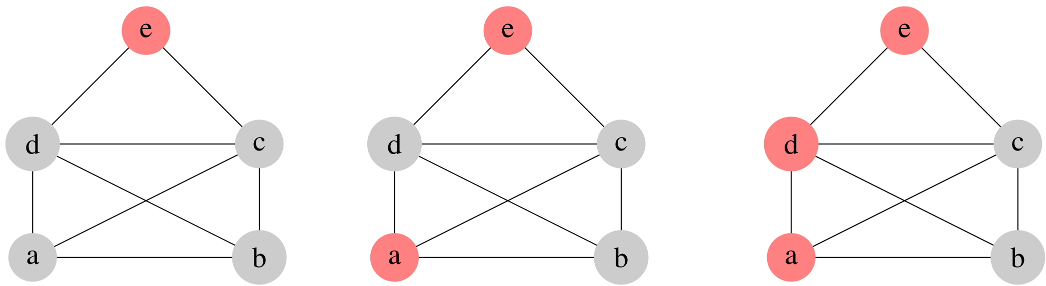
(b) Graf $P_5 \boxtimes P_3$

Gambar 2.3: Contoh graf hasil kali kartesius dan hasil kali kuat



(a) P_6^2 adalah graf kuasa ke-2 dari P_6 (b) C_{10}^3 adalah graf kuasa ke-3 dari C_3

Gambar 2.4: Contoh beberapa graf kuasa



(a) $S_1 = \{e\}$ (b) $S_2 = \{a, e\}$ (c) $S_2 = \{a, d, e\}$

Gambar 2.5: Mencari dimensi metrik graf amplop

2.3.3 Graf kuasa ke-k

Definisi 3 (Puš, 1991)**.** Graf G^k, disebut **graf kuasa ke-k dari graf** G, adalah graf dengan himpunan titik $V(G)$ dan untuk sebarang titik $u, v \in V(G^k)$ bertetangga jika dan hanya jika $d_G(u, v) \leq k$.

2.4 Dimensi Metrik

Misalkan G adalah graf terhubung. *Representasi metrik* titik v terhadap himpunan himpunan terurut $W = \{w_1, w_2, w_3, ..., w_k\} \subseteq V(G)$ adalah k-vektor terurut

$$r(v|W) = (d(v, w_1), d(v, w_2), ..., d(v, w_k)).$$

Himpunan W disebut *himpunan pembeda* bagi G jika tiap titik di G memiliki representasi yang unik. Himpunan pembeda yang mempunyai banyak titik paling sedikit disebut *basis metrik* bagi G, dengan kardinalitasnya disebut *dimensi metrik* G atau ditulis $dim(G)$.

Perhatikan G graf amplop pada Gambar 2.5. Terlihat bahwa jika dipilih $S_1 = \{e\}$, maka titik d dan c tidak terbedakan karena jaraknya ke S sama-sama 1. Kemudian, pemilihan $S_2 = \{a, e\}$ juga tidak bisa membedakan titik d dan c karena representasi metriknya sama-sama $(1, 1)$. Dapat ditunjukkan bahwa tidak ada himpunan pembeda dengan kardinalitas 1 atau 2 yang mampu untuk membedakan seluruh titik pada graf amplop tersebut. Dipilih $S_3 = \{a, d, e\}$ yang mampu membedakan semua titik pada graf tersebut. Representasi tiap titik-titiknya adalah:

$$r(a|S_3) = (0, 1, 2) \qquad r(c|S_3) = (1, 1, 1) \qquad r(e|S_3) = (2, 1, 0)$$
$$r(b|S_3) = (1, 1, 2) \qquad r(d|S_3) = (1, 0, 1)$$

Karena S_3 himpunan pembeda minimal, maka $pd(G) = |S_3| = 3$.

2.5 Dimensi Partisi

Π adalah partisi titik graf G yang berukuran k jika memenuhi definisi $\Pi = \{S_i \subset V(G) | \bigcup_{i=1}^{k} S_i = V(G), S_i \cap S_j = \emptyset, i \neq j, 1 \leq i, j \leq k\}$. Π disebut k-partisi karena berkadinalitas k.

Misalkan G adalah graf terhubung. Untuk sebuah titik $v \in V(G)$, representasi dari v terhadap k-partisi titik terurut $\Pi = \{S_1, S_2, ..., S_k\}$, didefinisikan sebagai k-vektor

$$r(v|\Pi) = (d(v, S_1), d(v, S_2), ..., d(v, S_k)).$$

Partisi Π disebut *partisi pembeda* jika setiap k-vektor $r(v|\Pi)$, $v \in V(G)$, unik. Partisi pembeda yang mempunyai banyak partisi yang paling sedikit disebut *basis* bagi G dengan kardinalitnya disebut *dimensi partisi* dari G atau ditulis $pd(G)$.

Perhatikan kembali G graf amplop pada Gambar 2.6. Dapat langsung dikonstruksikan partisi pembeda yang menggunakan setiap titik di himpunan pembeda minimumnya (Gambar 2.5c) sebagai partisi singleton dan titik sisanya sebagai satu partisi yang berbeda. Namun, ternyata G memiliki partisi pembeda yang ukurannya lebih kecil, seperti terlihat pada Gambar 2.6b.

(a) $\Pi_1 = \{\{a\}, \{d\}, \{e\}, \{b, c\}\}$ (b) $\Pi_2 = \{\{a, d\}, \{e\}, \{b, c\}\}$

Gambar 2.6: Berbagai basis partisi graf amplop

Berikut representasi titik-titik di G terhadap partisi Π_2.

$$r(a|\Pi_2) = (0, 2, 1) \qquad r(c|\Pi_2) = (1, 1, 0) \qquad r(e|\Pi_2) = (1, 0, 1)$$
$$r(b|\Pi_2) = (1, 2, 0) \qquad r(d|\Pi_2) = (0, 1, 1)$$

Dari hubungan antara dimensi metrik dan dimensi partisi tersebut, diperoleh Teorema 1 yang dijelaskan di subbab selanjutnya.

2.5.1 Hasil-Hasil Penelitian Dimensi Partisi

Teorema 1 (Chartrand et al., 2000)**.** *Untuk G suatu graf tak trivial, terhubung, dan sederhana,*

$$pd(G) \leq dim(G) + 1 \tag{2.5.1}$$

Bukti. Misalkan $dim(G) = k$ dan $S = \{v_1, v_2, ..., v_k\}$ adalah basis bagi G. Pandang partisi terurut $\Pi = \{S_1, S_2, ..., S_{k+1}\}$ dengan $S_i = \{v_i\}(1 \leq i \leq k)$ dan $S_{k+1} = V(G) - S$. Karena $r(u, S) = (d(u, v_1), d(u, v_2), ..., d(u, v_k), 0)$ untuk $u \in V(G) - S$ dan S adalah himpunan pembeda bagi G, maka representasi $r(u, \Pi), u \in S_{k+1}$ unik. Selain itu, hanya representasi $r(v_i|\Pi)$ yang punya nilai 0 pada entri ke-i, $1 \leq i \leq k$,. Artinya, $r(u|\Pi) \neq r(v_i|\Pi)$ untuk semua $u \in S_{k+1}$ dan semua $1 \leq i \leq k$. Akibatnya, Π adalah partisi pembeda bagi G, berukuran $k + 1$. Jadi, $pd(G) \leq dim(G) + 1$. ∎

Lemma 2 (Chartrand et al., 2000)**.** *Misalkan* Π *partisi pembeda bagi* G *dan* $u, v \in V(G)$*. Jika* $d(u, w) = d(v, w)$ *untuk semua* $w \in V(G) - \{u, v\}$*, maka* u *dan* v

berada pada di himpunan yang berbeda pada Π.

Bukti. Misalkan $\Pi = \{S_1, S_2, ..., S_k\}$ dengan $u, v \in S_i \in \Pi$. Maka $d(u, S_i) = d(v, S_i) = 0$. Karena $d(u, w) = d(v, w)$ untuk semua $w \in V(G) - \{u, v\}$, dan $d(u, S_j) = d(v, S_j)$ untuk seluruh $j, 1 \leq j \neq i \leq k$. Jadi, $r(u|\Pi) = r(v|\Pi)$ dan Π bukanlah partisi pembeda. ■

Teorema 3 (Chartrand et al., 2000)**.** *Jika G adalah graf dengan orde n (*≥ 3*) dan diameter d, maka*

$$g(n, d) \leq pd(G) \leq n - d + 1, \qquad (2.5.2)$$

dengan $g(n, d) = k$ *adalah bilangan bulat terkecil yang memenuhi* $(d + 1)^k \geq n$.

Bukti. Pandang G graf sederhana, terbatas, dan berdiameter d dengan $V(G) = \{v_1, v_2, ..., v_d, ...v_n\}$.

Pertama, akan buktikan batas atasnya. Misalkan u dan v adalah titik di G yang memenuhi $d(u, v) = d$ dan misalkan $u = v_1, v_2, ..., v_{d+1} = v$ adalah lintasan $u - v$ yang panjangnya d.

Konstruksi partisi

$$\Pi = \{S_1, S_2, ..., S_(n - d + 1)\}$$

dengan $S_1 = \{v_1, v_2, ..., v_d\}$ dan $S_i = \{v_(i + d - 1)\}$, untuk $2 \leq i \leq n - d + 1$.

Jelas bahwa titik-titik di S_1 terbedakan oleh S_2. Kemudian, S_i, untuk $2 \leq i \leq n-d+1$, adalah partisi singleton, sehingga semua titik yang berada di tiap partisi tersebut terbedakan. Didapat bahwa semua titik terbedakan. Artinya, Π adalah partisi pembeda sehingga $pd(G) \leq n - d + 1$.

Selanjutnya, perhatikan batas bawah. Misalkan $pd(G) = k$ dan Π adalah partisi pembeda berukuran k untuk $V(G)$. Perhatikan bahwa setiap representasi titik di G adalah tuple berukuran k dengan pilihan angka koordinatnya ada sebanyak $d + 1$ buah $(0, 1, .., d)$. Selain itu, kesemua n representasi haruslah unik. Akibatnya, k

haruslah bilangan bulat positif terkecil yang memenuhi $g(n, d) = (d + 1)^k \geq n$ sehingga $pd(G) = k \geq g(n, d)$.

Jadi, diperoleh $g(n, d) \leq pd(G) \leq n - d + 1$. ■

Teorema 4 (Chappell et al., 2008)**.** *Jika G graf dengan orde $n \geq 3$ dan diameter d, maka*

$$h(n, d) \leq pd(G) \leq n - d + 1, \tag{2.5.3}$$

dengan $h(n, d) = k$ adalah bilangan bulat terkecil yang memenuhi $k(d)^{k-1} \geq n$.

Bukti. Batas atas telah dibuktikan pada Teorema 3. Akan dibuktikan batas bawah yang dimodifikasi dari teorema teresebut.

Misalkan $pd(G) = k$ dan Π adalah partisi berkardinalitas k yang membedakan $V(G)$. Perhatikan bahwa setiap representasi titik di G adalah tuple berukuran k. Salah satu entrinya haruslah bernilai 0, menandakan urutan partisi yang mengandung titik tersebut. Terdapat sebanyak k pilihan tempat untuk entri 0. Selanjutnya, ada sebanyak d buah pilihan angka koordinat, yaitu $1, .., d$, untuk $k - 1$ entri yang tersisa. Oleh karena itu, terdapat paling banyak kd^{k-1} kemungkinan representasi titik sedemikian hingga keseluruh n titik di G terbedakan. Akibatnya, k haruslah bilangan bulat positif terkecil yang memenuhi $kd^{k-1} \geq n$ sehingga $pd(G) \geq k = h(n, d)$. ■

Teorema 4 memperbaiki Teorema 3 karena batas bawahnya lebih ketat. Diketahui kesamaan dengan batas bawah tersebut dipenuhi oleh graf lintasan dan graf lengkap.

Berikut adalah beberapa karakterisasi dimensi partisi paling sederhana.

Teorema 5 (Chartrand et al., 2000)**.** *Misalkan n adalah orde graf, maka*

$$pd(G) = 2 \Leftrightarrow G = P_n \tag{2.5.4}$$

$$pd(G) = n \Leftrightarrow G = K_n \tag{2.5.5}$$

Hubungan antara diameter, orde, dan dimensi partisi suatu graf ditunjukkan oleh teorema berikut.

Teorema 6 (Chappell et al., 2008)**.** *Orde maksimum dari graf berdiameter 2 dan berdimensi partisi $k \geq 2$ adalah*

$$l\left[\binom{2l-1}{l} + 2^{2l-1}\right],\ jika\ k = 2l$$

dan

$$(2l+1)\left[\binom{2l-1}{l} + 2^{2l-1}\right],\ jika\ k = 2l+1.$$

Berikut beberapa nilai eksak dimensi partisi beberapa kelas graf yang ikut dikaji.

Akibat 6.1 (Chartrand et al., 2000)**.**

$$pd(C_n) = 3 \tag{2.5.6}$$

$$pd(K_{r,s}) = \begin{cases} r+1, & r = s \\ max\{r,s\}, & r \neq s \end{cases} \tag{2.5.7}$$

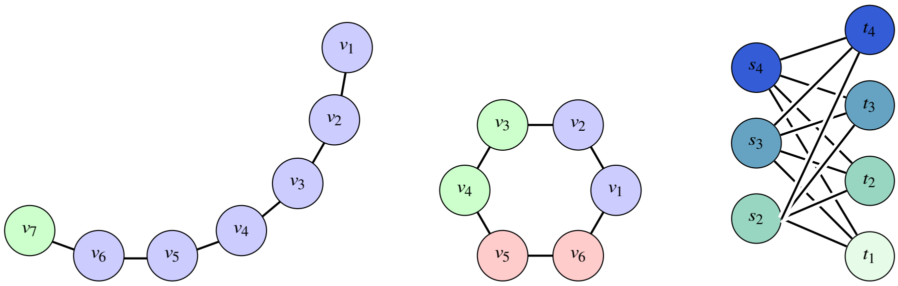
Contoh 2. $\Pi = \{\{v_1, ..., v_6\}, \{v_7\}\}$ adalah salah satu partisi pembeda minimal untuk P_7. Berikut representasi tiap titiknya.

$$r(v_7|\Pi) = (1,0) \qquad r(v_i|\Pi) = (0, 7-i), 1 \leq i \leq 6$$

Contoh 3. $\Pi = \{\{v_1, v_2\}, \{v_3, v_4\}, \{v_5, v_6\}\}$ adalah salah satu partisi pembeda minimal untuk C_6. Berikut representasi tiap titiknya.

$$r(v_1|\Pi) = (0,2,1) \qquad r(v_3|\Pi) = (1,0,2) \qquad r(v_5|\Pi) = (2,1,0)$$
$$r(v_2|\Pi) = (0,1,2) \qquad r(v_4|\Pi) = (2,0,1) \qquad r(v_6|\Pi) = (1,2,0)$$

Contoh 4. $\Pi = \{\{t_1\}\{t_2, s_2\}, \{t_3, s_3\}, \{t_4, s_4\}\}$ adalah salah satu partisi pembeda



(a) P_7 (b) C_6 (c) $K_{3,4}$

minimal untuk $K_{3,4}$. Berikut representasi tiap titiknya.

$$r(t_1|\Pi) = (0, 1, 1, 1)$$

$$r(t_2|\Pi) = (2, 0, 1, 1) \qquad r(t_3|\Pi) = (2, 1, 0, 1) \qquad r(t_4|\Pi) = (2, 1, 1, 0)$$

$$r(s_2|\Pi) = (2, 0, 1, 1) \qquad r(s_3|\Pi) = (1, 1, 0, 1) \qquad r(s_4|\Pi) = (1, 1, 1, 0)$$

Bab 3 Graf Hasil Kali 2-Kuat dan Dimensi Partisinya

3.1 Graf Hasil Kali k-Kuat

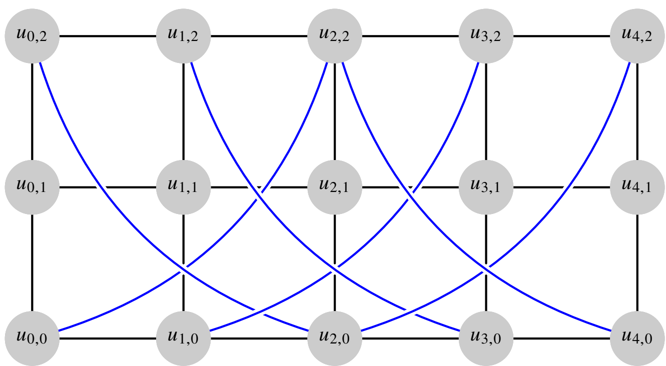
Perhatikan bahwa pada definisi 2, syarat ke-tiga sisi yang dibentuk adalah sisi dari titik-titik yang saling bertetangga di graf asalnya. Kita dapat memandang titik-titik tersebut sebagai sepasang yang berjarak 1. Dari situ, kita perumum definisi operasi kali kuat dengan syarat ketiga yang dimodifikasi.

Definisi 4. Untuk graf G dan H, **graf hasil kali k-kuat** $G \boxtimes_k H$ adalah graf yang memiliki himpunan titik $V(G) \times V(H)$, dengan titik (u, x) dan (v, y) bertetangga jika dan hanya jika

- $uv \in E(G)$ dan $x = y$, atau
- $u = v$ dan $xy \in E(H)$, atau
- $d_G(u, v) = k$ dan $d_H(x, y) = k$.

Sisi yang dibentuk oleh poin terakhir dinamakan sebagai **sisi kuat** dan titik-titik yang terkait dengannya disebut **titik kuat**.

Jika tidak ada sepasang titik baik di graf G maupun graf H yang berjarak lebih dari k, maka graf hasil kali k-kuat antara dua graf tersebut akan sama saja dengan graf hasil kali Kartesiusnya. Dari sini, diformulasikan syarat diameter graf komponen dari graf hasil kali k-kuat.



Gambar 3.1: Graf $P_5 \boxtimes_2 P_3$

Proposisi 1. *Misalkan G dan H adalah graf dengan minimal diameternya kurang dari $k(k \geq 2)$, maka*

$$G \boxtimes_k H = G \square H.$$

Bukti. Misalkan $min\{diam(G), diam(H)\} < k$. Jelas terlihat dari definisi graf hasil kali Kartesius, bahwa tidak terdapat pasangan titik di graf G dan H yang memenuhi syarat ketiga definisi graf hasil kali k-kuat, karena $d_{G \square H}(u, v) < k$, sehingga $G \boxtimes_n H$ tidak memiliki sisi kuat. ■

Dari Proposisi 1, kita batasi bahasan kita pada graf-graf yang berdiameter minimal k untuk mengkaji graf hasil kali k-kuat. Dari syarat diameter tersebut, cukup jelas bahwa graf hasil kali k-kuat akan mengandung subgraf hasil kali k-kuat lintasan berorde $k + 1$.

Observasi 1. *Misalkan G dan H adalah graf dengan diameter terkecilnya k, maka graf $G \boxtimes_k H$ mengandung subgraf yang isomorfis terhadap $P_{k+1} \boxtimes P_{k+1}$.*

Jarak antar titik pada graf kali 2-kuat dapat diketahui nilainya secara pasti untuk titik-titik pada lapisan yang sama. Dua titik berbeda $(a, b), (\alpha, \beta) \in V(G \boxtimes_2 H)$ berada di lapisan-G jika $a = \alpha$ dan berada di lapisan-H jika $b = \beta$.

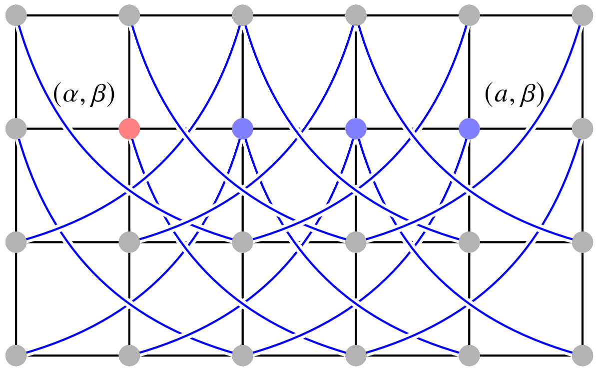
Lemma 7. *Misalkan G dan H graf sederhana, terhubung, tak-trivial, dengan $min\{diam(G), diam(H)\} \geq 2$, serta titik $u = (a, b) \in V(G \boxtimes_2 H)$. Misalkan pula $v = (\alpha, \beta) \in V(G \boxtimes_2 H)$.*

- *Jika $\beta = b$, maka*

$$d_{G \boxtimes_2 H}(u, v) = 2 \left\lfloor \frac{d_G(a, \alpha)}{4} \right\rfloor + (d_G(a, \alpha) \mod 4) \tag{3.1.1}$$

- *Jika $\alpha = a$, maka*

$$d_{G \boxtimes_2 H}(u, v) = 2 \left\lfloor \frac{d_H(b, \beta)}{4} \right\rfloor + (d_H(b, \beta) \mod 4). \tag{3.1.2}$$



Gambar 3.2: Ilustrasi Lemma 7

Bukti. Dalam mencari lintasan terpendek antara dua titik pada graf hasil kali 2-kuat, kita dapat memilih strategi yang mengutamakan sisi-sisi kuat dilewati terlebih dahulu. Perhatikan bahwa lintasan yang berasal dari titik u akan dapat kembali ke baris u jika telah melewati 2 sisi kuat dan kelipatannya. Sementara itu, 2 lompatan yang dilakukan 2 sisi kuat akan melewati empat titik di baris u. Dengan begitu, kita wakilkan langkah ini dengan $2\left\lfloor \frac{d_G(a,\alpha)}{4} \right\rfloor$.

Kemudian, ketika setelah menggunakan sisi-sisi kuat lintasan sudah cukup dekat dengan titik v, kita akan melintasi sisi-sisi di baris u untuk mencapai v. Jika langkah-langkah sebelumnya telah optimum, maka sisi yang harus dilewati dapat sebanyak 0, 1, 2, atau 3, bergantung seberapa besar jarak v dengan titik kuat terdekatnya. Langkah ini diwakilkan oleh $(d_H(b,\beta) \mod 4)$. Sehingga kita dapatkan persamaan yang diinginkan. ■

3.2 Graf Kuasa Ke-k dari Suatu Graf

Secara umum, graf hasil kali k-kuat memiliki properti yang cukup kompleks. Oleh karena itu, untuk meneliti graf hasil kali 2-kuat, akan dimanfaatkan kemiripannya dengan graf kuadrat (graf kuasa ke-2) dari setiap komponen grafnya. Di akhir sub bab ini, diperoleh batas atas dari dimensi partisi graf hasil kali 2-kuat yang dijelaskan oleh dimensi partisi graf-graf komponennya.

Pertama, dari definisi graf kuasa ke-k suatu graf, dapat dengan mudah diketahui jarak antar titik pada graf kuadrat.

Lemma 8. *Misalkan G suatu graf sederhana, terbatas, dan terhubung, dengan $u, v \in V(G) = V(G^2)$, maka*

$$d_{G^2}(u, v) = \left\lceil \frac{d_G(u, v)}{2} \right\rceil.$$

Bukti. Dari definisi graf kuasa, kita punya graf G^2 adalah graf dengan $V(G^2) = V(G)$ dan $E(G^2) = E(G) \cup \{ab | a, b \in V(G), d_G(a, b) = 2\}$. Artinya, untuk setiap pasang titik u, v, terdapat lintasan pada G^2 yang menghubungkan u dan v yang panjangnya lebih pendek daripada lintasan terpendek pada G yang menghubungkan u dan v. Tepatnya, lintasan pada G^2 tersebut akan melalui sisi-sisi yang berasal dari $\{ab | a, b \in G, d_G(a, b) = 2\}$. Sehingga kita dapatkan persamaan di atas. ∎

Selanjutnya, Lemma 8 dapat dimanfaatkan untuk mengidentifikasi karakter titik-titik pada suatu graf, yang dengan partisi pembeda yang sama, mereka tetap terbedakan pada graf kuadratnya.

Lemma 9. *Misalkan G graf sederhana, terbatas, dan terhubung dengan $diam(G) \geq 2$ dan Π adalah partisi pembedanya. Jika u, v tidak bertetangga di G, maka u, v di G^2 juga masih terbedakan oleh Π.*

Bukti. Misalkan Π partisi pembeda bagi G. Misalkan pula u, v tidak bertetangga di G, maka $d_G(u, v) = s \geq 2$. Pandang sebarang himpunan $A \in \Pi$. Misalkan $d_G(v, A) = t$, maka $d_G(u, A) = s + t$. Perhatikan bahwa $d_{G^2}(u, A) = \left\lceil \frac{s+t}{2} \right\rceil$ dan $d_{G^2}(v, A) = \left\lceil \frac{t}{2} \right\rceil$. Karena $s \geq 2$, dengan induksi, jelas bahwa $\left\lceil \frac{s+t}{2} \right\rceil > \left\lceil \frac{t}{2} \right\rceil \implies d_{G^2}(u, A) \neq d_{G^2}(v, A)$. Jadi, u, v terbedakan oleh Π. ∎

3.2.1 Dimensi Partisi Graf Kuadrat

Dengan diketahuinya sifat dari Lemma 9, dapat disusun strategi modifikasi partisi pembeda suatu graf sehingga graf kuadratnya tetap terbedakan.

Teorema 10. *Misalkan G suatu graf sederhana, terbatas, dan terhubung dengan*

$diam(G) \geq 2$, *maka*

$$pd(G^2) \leq pd(G) + 2^{pd(G)-1} - 1. \quad (3.2.1)$$

Bukti. Misalkan $\Pi = \{\Pi_1, \Pi_2, ..., \Pi_{pd(G)}\}$ adalah partisi pembeda dari G. Misalkan titik $u \in A_i, 1 \leq i \leq pd(G)$, dengan representasinya terhadap Π di graf G adalah

$$r(u|\Pi) = (d_1, d_2, ..., d_{pd(G)}),$$

dengan $d_i = 0$.

Berdasarkan Lemma **??**, representasi u terhadap Π di graf G^2 adalah

$$r(u|\Pi) = (\delta_1, \delta_2, ..., \delta_{pd(G)}),$$

dengan $\delta_i = \lceil d_i/2 \rceil, 1 \leq i \leq pd(G)$.

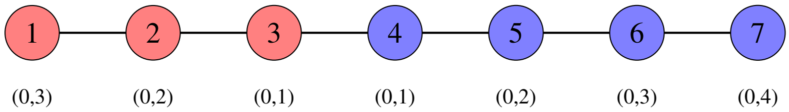
Oleh karena itu, terdapat titik-titik di graf G^2 yang representasi terhadap Π-nya sama. Kemudian, karena untuk setiap i, $1 \leq i \leq pd(G)$, δ_i adalah hasil pemetaan dari dua nilai (d_i ganjil dan $d_i + 1$), maka dalam suatu himpunan A_i, maksimal banyaknya titik dengan suatu representasi yang sama adalah $2^{pd(G)-1}$.

Misalkan G^2 memiliki sebanyak n himpunan titik berepresentasi sama tadi yaitu, $R_m, (m = 1, ..., n)$ yang masing-masing berukuran $s_m \leq 2^{pd(G)-1}$. Misalkan $s = max\{s_1, ..., s_m\}$. Definisikan himpunan $R_m = \{u_{m,1}, ..., u_{m,s_m}\}$ untuk setiap m, sedemikian hingga $u_{a,c}$ dan $u_{b,c}$ tidak bertetangga, untuk setiap $a, b \in \{1, ..., n\}, a \neq b$, dan setiap $c \in \{1, ..., s\}$.

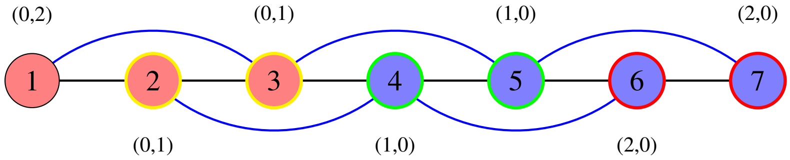
Selanjutnya, kita konstruksi koleksi himpunan baru $B = \{B_1, ..., B_{s-1}\}$, dengan

$$B_i = \{u_{m,i} | 1 \leq m \leq n\}.$$

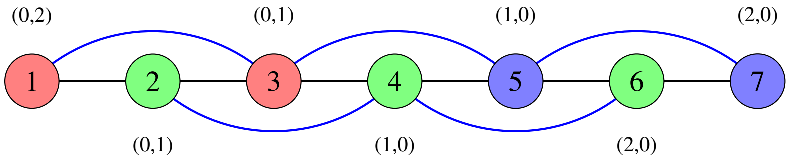
Misalkan $\Pi' = \{\Pi_i - B_i\}$. Karena Π adalah partisi pembeda bagi G maka Π' membedakan seluruh titik $u \in \Pi'_j, \forall j$. Kemudian, B membedakan seluruh pasangan



(a) P_7 dengan partisi Π



(b) P_7^2 dengan partisi Π



(c) P_7^2 dengan partisi $\Pi' \cap B$

Gambar 3.3: Ilustrasi Contoh 5

$u, v \in B_i, \forall i$, karena u, v tidak bertetangga (Lemma 9). Dapat disimpulkan bahwa partisi $\Pi' \cup B$ membedakan $V(G^2)$.

Karena $|B| = s - 1 \leq 2^{pd(G)-1} - 1$, maka $pd(G^2) \leq pd(G) + 2^{pd(G)-1} - 1$. ■

Contoh 5. Pandang graf P_7 dan P_7^2 dengan $V(P_7) = V(P_7^2) = \{i : 1 \leq i \leq 7,\}$. Definisikan partisi $\Pi = \{\{1, 2, 3\}, \{4, 5, 6, 7\}\}$ sehingga P_7 terbedakan oleh Π seperti yang terlihat pada Gambar 3.3a. Namun Π tidak membedakan P_7^2. Terdapat tiga himpunan titik berepresentasi sama, yaitu $R_1 = \{2, 3\}$ dengan representasi (0,1), $R_2 = \{4, 5\}$ dengan representasi (1,0), dan $R_3 = \{6, 7\}$ dengan representasi (2,0). Karena kardinalitas terbesar dari R_i tadi adalah 2, kebetulan sama dengan $2^{pd(P_7)-1}$, maka cukup kita konstruksikan $2 - 1 = 1$ sel partisi tambahan yang anggotanya adalah satu titik dari tiap himpunan titik berepresentasi sama. Pilih $B_1 = \{2, 4, 6\}$ sehingga kita punya $\Pi' = \{\{1, 3\}, \{5, 7\}\}$. Kita dapatkan $\Pi' \cup B_1$ membedakan P_7^2 seperti terlihat pada Gambar 3.3c.

Sejauh ini, diketahui batas atas Teorema 10 ketat untuk graf lintasan. Batas tersebut akan membantu untuk graf-graf yang dimensi partisinya tidak bergantung terhadap ordenya, seperti graf lintasan dan graf lingkaran.

Kemudian di bawah ini ditunjukkan bahwa jarak antar titik di graf hasil kali 2-kuat dibatasi oleh jarak antar titik di graf kuadrat graf-graf komponennya. Keterangan ini akan berguna untuk mendapatkan hubungan dimensi partisi antara kedua jenis graf tersebut.

3.2.2 Hubungan Dimensi Partisi Graf Kali 2-Kuat dan Graf Kuadrat

Lemma 11. *Misalkan G dan H adalah graf terhubung non trivial yang memenuhi* $min\{diam(G), diam(H)\} \geq 2$*. Misalkan* $A \subset V(G)$ *dan* $B \subset V(H)$*.*

Jika $a \in A$ *dan* $b \notin B$*, maka*

$$d_{H^2}(b, B) \leq d_{G \boxtimes_2 H}((a, b), A \times B) \leq d_{H^2}(b, B) + 2. \tag{3.2.2}$$

Jika $a \notin A$ *dan* $b \in B$*, maka*

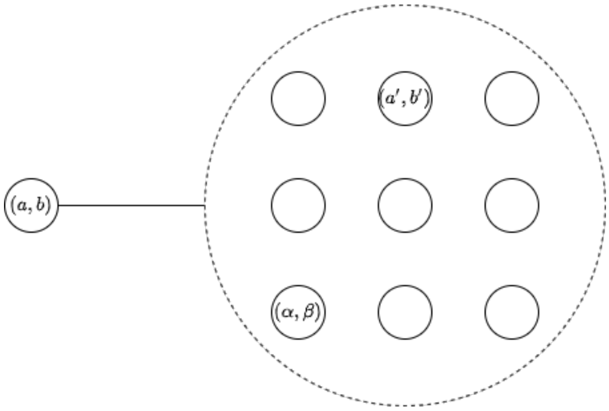
$$d_{G^2}(a, A) \leq d_{G \boxtimes_2 H}((a, b), A \times B) \leq d_{G^2}(a, A) + 2. \tag{3.2.3}$$

Bukti. Pandang $a \in A$ dan $b \notin B$. Misalkan $d_{H^2}(b, B) = s$. Karena graf hasil kali 2-kuat mengandung sisi-sisi kuat, kita klaim bahwa terdapat subgraf $I = P_3 \boxtimes_2 P_3$ di $G \boxtimes_2 H$ sedemikian sehingga I memuat $(a', b') \in A \times B$ dan memuat (α, β), dengan $d_{G \boxtimes_2 H}((a, b), (a', b')) = d_{G \boxtimes_2 H}((a, b), A \times B)$ dan $d_{G \boxtimes_2 H}((a, b), (\alpha, \beta)) = s$.

Karena $diam(P_3 \boxtimes_2 P_3) = 2$, maka $d_{G \boxtimes_2 H}((\alpha, \beta), (a', b')) \leq 2$. Sehingga kita punya $s \leq d_{G \boxtimes_2 H}((a, b), A \times B) \leq s + 2$ atau persamaan 3.2.2. Persamaan didapat jika $(\alpha, \beta) = (a', b')$. Lalu, dengan cara serupa, untuk $a \notin A$ dan $b \in B$, kita dapatkan persamaan 3.2.3 ■

Teorema 12. *Untuk graf-graf sederhana, terbatas, dan terhubung G dan H dengan* $min\{diam(G), diam(H)\}) \geq 2$*,*

$$pd(G \boxtimes_2 H) \leq pd(G^2) \cdot pd(H^2). \tag{3.2.4}$$



Gambar 3.4: Ilustrasi Lemma 11

Bukti. Misalkan $\Pi_1 = \{A_1, A_2, ..., A_s\}$ dan $\Pi_2 = \{B_1, B_2, ..., B_t\}$ masing-masing adalah partisi pembeda dari G^2 dan H^2. Akan ditunjukkan bahwa $\Pi = \{A_i \times B_j : 1 \leq i \leq s, 1 \leq j \leq t\}$ adalah himpunan pembeda dari $G \boxtimes_2 H$.

Misalkan $(a, b), (\alpha, \beta)$ adalah dua titik berbeda dari $V(G \boxtimes_2 H)$, dengan $a, \alpha \in V(G) = V(G^2)$ dan $b, \beta \in V(H) = V(H^2)$. Perhatikan bahwa jika $(a, b), (\alpha, \beta)$ berada pada himpunan yang berbeda di Π, maka kedua titik tersebut terbedakan oleh Π, karena $d_{G^2}(a, A_i) \neq d_{G^2}(\alpha, A_i)$, untuk $1 \leq i \leq s$; dan $d_{H^2}(b, B_i) \neq d_{G^2}(\beta, B_i)$, untuk $1 \leq i \leq t$.

Selanjutnya kita tinjau kasus $(a, b), (\alpha, \beta)$ berada pada himpunan yang sama di Π.

- **Kasus 1:** $a = \alpha$, maka terdapat $i \in \{1, ..., s\}$ sedemikian hingga $a \in A_i$. Selain itu, perhatikan bahwa terdapat $B_j \in \Pi_2$ untuk suatu $j \in \{1, ..., t\}$, sedemikian sehingga $d_{H^2}(b, B_j) \neq d_{H^2}(\beta, B_j\}$

  $\implies d_{H^2}(b, B_j) - d_{H^2}(\beta, B_j\} \neq 0.$

  Dari Lemma 11, kita punya

  $$d_{H^2}(b, B_j) \leq d_{G \boxtimes_2 H}((a, b), A_i \times B_j) \leq d_{H^2}(b, B_j) + 2$$

  dan

  $$d_{H^2}(\beta, B_j) \leq d_{G \boxtimes_2 H}((\alpha, \beta), A_i \times B_j) \leq d_{H^2}(\beta, B_j) + 2$$

Dengan mencari selisih dua pertidaksamaan di atas, diperoleh

$d_{G\boxtimes_2 H}((a,b), A_i \times B_j) - d_{G\boxtimes_2 H}((\alpha,\beta), A_i \times B_j) = d_{H^2}(b, B_j) - d_{H^2}(\beta, B_j)$

$\implies d_{G\boxtimes_2 H}((a,b), A_i \times B_j) - d_{G\boxtimes_2 H}((\alpha,\beta), A_i \times B_j) \neq 0$

$\implies d_{G\boxtimes_2 H}((a,b), A_i \times B_j) \neq d_{G\boxtimes_2 H}((\alpha,\beta), A_i \times B_j).$

$\implies (a,b), (\alpha,\beta)$ terbedakan.

- **Kasus 2:** $a \neq \alpha$, maka terdapat $A_k \in \Pi_1$ untuk suatu $k \in \{1, ..., t\}$, sedemikian sehingga $d_{G^2}(a, A_k) \neq d_{G^2}(\alpha, A_k\}$. Selain itu, karena $(a,b), (\alpha,\beta)$ berada dalam himpunan yang sama di Π, terdapat $l \in \{1, ..., t\}$ sedemikian hingga $b, \beta \in B_l$. Serupa dengan argumen sebelumnya, dengan menggunakan Lemma 11, kita punya

$$d_{G^2}(a, A_k) \leq d_{G\boxtimes_2 H}((a,b), A_k \times B_l) \leq d_{G^2}(a, A_k) + 2$$

dan

$$d_{G^2}(\alpha, A_k) \leq d_{G\boxtimes_2 H}((a,b), A_k \times B_l) \leq d_{G^2}(\alpha, A_k) + 2.$$

Dengan mencari selisih dua pertidaksamaan di atas, diperoleh

$d_{G\boxtimes_2 H}((a,b), A_k \times B_l) - d_{G\boxtimes_2 H}((\alpha,\beta), A_k \times B_l) = d_{G^2}(a, A_k) - d_{G^2}(\alpha, A_k)$

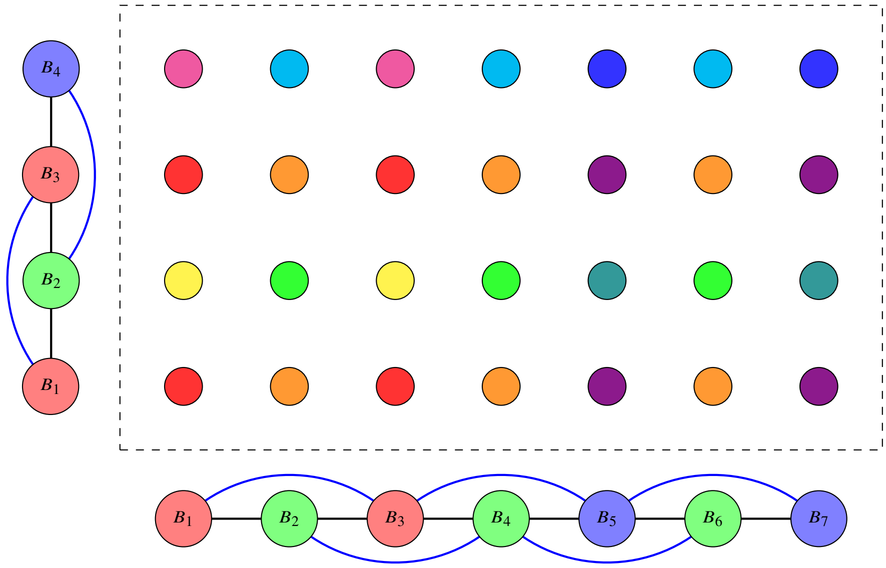
$\implies d_{G\boxtimes_2 H}((a,b), A_k \times B_l) - d_{G\boxtimes_2 H}((\alpha,\beta), A_k \times B_l) \neq 0$

$\implies d_{G\boxtimes_2 H}((a,b), A_k \times B_l) \neq d_{G\boxtimes_2 H}((\alpha,\beta), A_k \times B_l).$

$\implies (a,b), (\alpha,\beta)$ terbedakan.

Akibatnya, untuk setiap dua titik berbeda $(a,b), (\alpha,\beta) \in V(G \boxtimes_2 H)$, didapat $r((a,b)|\Pi) \neq r((\alpha,\beta)|\Pi)$ sehingga Π himpunan pembeda dan batas atas terpenuhi.

■



Gambar 3.5: Ilustrasi batas atas Teorema 12, partisi pembeda untuk graf $P_4 \boxtimes_2 P_7$

3.3 Dimensi Partisi Graf Hasil Kali 2-Kuat Beberapa Kelas Graf

Observasi 2. *Untuk G dan H graf sederhana, terhubung, dan berdiameter minimal 2, diameter graf $G \boxtimes_2 H$ bergantung pada diameter paling besar antara diameter G dan H.*

3.3.1 Dimensi Partisi Graf Lintasan 2-Kuat Graf Lintasan

Observasi 3.

$$diam(P_m \boxtimes_2 P_n) = \begin{cases} \left\lceil \frac{n-1}{2} \right\rceil + 1, & m = 3, n \in \{3,4\} \text{ } atau \text{ } 4 \leq m \leq n \\ \left\lceil \frac{n-1}{2} \right\rceil + 2, & 4 \leq m \leq n \end{cases}$$

Misalkan graf P_r, memiliki himpunan sisi $\{0, 1, 2, ..., n-1\}$ dengan lintasan terpanjang adalah $0, 1, ..., n-2, n-1$. Perhatikan bahwa diameter dari graf lintasan berorde n adalah $n-1$, sehingga berdasarkan Lemma 8, diameter dari P_n^2 adalah $\left\lceil \frac{n-1}{2} \right\rceil$. Kemudian jarak antara $(0,0)$ dengan $(0, n-1)$ dan dengan $(2, n-1)$ dapat dipandang sebagai jarak antara titik 0 dengan titik $n-1$ pada graf P_n^2, dengan jarak yang mungkin adalah $\left\lceil \frac{n-1}{2} \right\rceil$ atau $\left\lceil \frac{n-1}{2} \right\rceil + 1$.

Dapat dicek dengan mudah bahwa diameter $P_3 \boxtimes_2 P_4$ dan $P_3 \boxtimes_2 P_4$ masing-masing adalah 2 dan 3. Selain itu, kita bagi menjadi beberapa kasus.

(i) $n \mod 4 = 3$. *Jika* $m \geq 4$, *maka lintasan untuk jarak terpanjangnya ialah (0,0),(2,2),...,(2,n-1), (1,n-1), sehingga diameternya* $\lceil \frac{n-1}{2} \rceil + 1$. *Jika* $m = 3, n \neq 3$, *maka maka lintasan untuk jarak terpanjangnya ialah (0,1),(0,0),(2,2),...,(2,n-1),(1,n-1) sehingga diameternya* $\lceil \frac{n-1}{2} \rceil + 1 = \lceil \frac{n-1}{2} \rceil + 2$.

(ii) $n \mod 4 = 0$. *Jika* $m \geq 4$, *maka lintasan untuk jarak terpanjangnya ialah (0,0),(2,2),...,(2,n-2),(2,n-1),(1,n-1), sehingga diameternya* $\lceil \frac{n-1}{2} \rceil + 1$. *Jika* $m = 3, n \neq 4$, *maka lintasan untuk jarak terpanjangnya ialah (0,1),(0,0),(2,2),...,(2,n-2),(2,n-1),(1,n-1), sehingga diameternya* $\lceil \frac{n-1}{2} \rceil + 2$.

(iii) $n \mod 4 = 1$. *Jika* $m \geq 4$ *maka lintasan untuk jarak terpanjangnya ialah (0,0),(2,2),...,(0,n-1),(1,n-1), sehingga diameternya* $\lceil \frac{n-1}{2} \rceil + 1$. *Jika* $m = 3$ *maka lintasan untuk jarak terpanjangnya adalah (0,1),(0,0),(2,2),...,(0,n-1),(1,n-1), sehingga diameternya* $1 + \lceil \frac{n-1}{2} \rceil + 1 = \lceil \frac{n-1}{2} \rceil + 2$.

(iv) $n \mod 4 = 2$. *Jika* $m \geq 4$ *maka lintasan untuk jarak terpanjangnya adalah (0,0),(2,2),...,(0,n-2),(0,n-1),(1,n-1), sehingga diameternya* $\lceil \frac{n-1}{2} \rceil + 1$. *Jika* $m = 3$ *maka lintasan untuk jarak terpanjangnya ialah (0,1),(0,0),(2,2),...,(0,n-2),(0,n-1),(1,n-1), sehingga diameternya* $1 + \lceil \frac{n-1}{2} \rceil + 1 = \lceil \frac{n-1}{2} \rceil + 2$.

Proposisi 2.

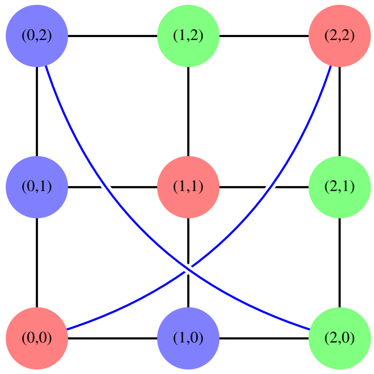
$$pd(P_3 \boxtimes_2 P_3) = 3 \tag{3.3.1}$$

Bukti. Misalkan $G = H = P_3$. Karena graf $G \boxtimes_2 H$ bukan graf lintasan, berdasarkan Akibat 5, maka haruslah $pd(G \boxtimes_2 H) \geq 3$. Misalkan $V(G) = V(H)\{0, 1, 2\}$, dapat dikonstruksi partisi pembeda $\Pi = \{A, B, C\}$ dengan $A = \{(0, 0), (1, 1), (2, 2)\}$, $B = \{(1, 0), (0, 1), (0, 2)\}$, $C = \{(2, 0), (2, 1), (1, 2)\}$ seperti terlihat pada Gambar 3.6. Dapat dibuktikan bahwa partisi berukuran 3 yang dapat membedakan graf ini hanyalah partisi yang isomorfis dengan Π. ■

Proposisi 3. *Untuk* $n \geq 4$,

$$pd(P_3 \boxtimes_2 P_n) \leq 4. \tag{3.3.2}$$

Bukti. Misalkan $S = \{(0, 0), (0, 2), (1, 1)\} \subset V(P_3 \boxtimes_2 P_n)$ dengan $V(P_3) = \{0, 1, 2\}$ dan $V(P_n) = \{0, 1, \ldots, n\}$. Perhatikan bahwa representasi titik-titik $(1, i)$, un-



Gambar 3.6: Partisi pembeda minimal untuk graf $P_3 \boxtimes_2 P_3$

tuk $0 \leq i \leq n$, terhadap S adalah (α, α, β), dengan $\alpha = \left\lceil \frac{d_{P_n}(i,1)}{2} \right\rceil + 1$ dan $\beta = d((1,i),\{1,1\}) = 2\left\lfloor \frac{d_{P_n}(i,1)}{4} \right\rfloor + (d_{P_n}(i,1) \mod 4)$.

Perhatikan pula bahwa karena kesimetrisan, representasi titik-titik $(0,i)$ dan $(2,i)$ terhadap S masing-masing adalah (a,b,c) dan (b,a,c), dengan c tertentu, dan $a = 2\left\lfloor \frac{d_{P_n}(i,0)}{4} \right\rfloor + (d_{P_n}(i,0) \mod 4)$ serta $b = 2\left\lfloor \frac{d_{P_n}(i,2)}{4} \right\rfloor + (d_{P_n}(i,2) \mod 4)$.

Jelas bahwa $a \neq b$. Akibatnya, semua titik $(0,i),(1,i),(2,i)$ terbedakan oleh S. Didapatkan $md(P_3 \boxtimes_2 P_n) \leq 3$, sehingga menurut Teorema 1,

$$pd(P_3 \boxtimes_2 P_n) \leq md(P_3 \boxtimes_2 P_n) + 1 \leq 4.$$

■

Dari observasi di atas, kita dapatkan batas bawah partisi dimensi graf lintasan 2-kuat graf lintasan adalah 3. Dari Teorema 12, karena $pd(P_n) = 3$, maka kita dapat kan batas atas dari graf ini adalah $3 \cdot 3 = 9$.

Akibat 12.1. *Untuk $m,n \geq 3$,*

$$3 \leq pd(P_m \boxtimes_2 P_n) \leq 9.$$

3.3.2 Dimensi Partisi Graf Lintasan 2-Kuat Graf Bipartit Lengkap

Observasi 4.

$$diam(P_n \boxtimes_2 K_{s,t}) = \begin{cases} n-1, & s = 1, t \geq 3, n \in \{3,4\} \\ \left\lceil \frac{n-1}{2} \right\rceil + 2, & s = 1, t \geq 3, n \geq 5 \\ \left\lceil \frac{n-1}{2} \right\rceil + 1, & s \geq 2, t \geq 3, n \geq 3 \end{cases}$$

Misalkan $V(K_{s,t}) = \{u_1, ..., u_s, v_1, ..., v_t\}$ *dengan partisi pertama adalah* $u_1, ...u_s$ *dan partisi kedua adalah* $v_1, ..., v_t$. *Misalkan pula* $V(P_n) = \{0, 1, 2, ..., n-1\}$.

Untuk $s = 1, t \geq 3, n \in \{3,4\}$, *perhatikan bahwa salah salah satu pasang titik dengan jarak terpanjang adalah titik* $(0, t_1)$ *dan titik* $(n-1, t_2)$, *dengan jaraknya ialah* $n-1$.

Untuk kasus kedua dan ketiga, kita dapat memilih lintasan untuk jarak terpanjang di graf lintasan 2-kuat graf bipartit lengkap dengan cara yang mirip dilakukan pada Observasi 3 pada subgraf yang isomorfis dengan $P_3 \boxtimes_2 P_n$.

Akibat 12.2.

$$4 \leq pd(P_3 \boxtimes_2 K_{s,t}) \leq 3(s+t) - 1 \qquad\qquad s \geq 1, t \geq 3$$

Contoh 6. Misalkan $V(P_n) = \{i : 0 \leq i \leq n-1\}$, dan $V(K_{s,t}) = \{i : 0 \leq i \leq s-1\} \cup \{i : s \leq i \leq s+t-1\}$. Untuk graf $P_3 \boxtimes_2 K_{1,3}$, pilih partisi pembeda

$$\Pi_1 = \{(0,0), (0,2), (1,0), (1,1)\},$$
$$\Pi_2 = \{(0,1), (0,3)\},$$
$$\Pi_3 = \{(1,2), (1,3), (2,0), (2,1), (2,2)\},$$
$$\Pi_4 = \{(2,3)\}.$$

Sehingga didapat representasi tiap titik sebagai berikut.

$$r((0,0)|\Pi) = (0,1,2,2) \qquad r((1,2)|\Pi) = (1,2,0,2)$$
$$r((0,2)|\Pi) = (0,2,1,1) \qquad r((1,3)|\Pi) = (1,1,0,1)$$
$$r((1,0)|\Pi) = (0,2,1,2) \qquad r((2,0)|\Pi) = (1,2,0,1)$$
$$r((1,1)|\Pi) = (0,1,1,2) \qquad r((2,1)|\Pi) = (1,1,0,2)$$
$$r((0,1)|\Pi) = (1,0,1,1) \qquad r((2,2)|\Pi) = (2,1,0,2)$$
$$r((0,3)|\Pi) = (1,0,1,2) \qquad r((2,3)|\Pi) = (1,1,1,0)$$

Jadi $pd(P_3 \boxtimes_2 K_{1,3} = 4$

3.3.3 Dimensi Partisi Graf Lingkaran 2-Kuat Graf Bipartit Lengkap

Observasi 5.

$$diam(C_n \boxtimes_2 K_{s,t}) = \begin{cases} \left\lfloor \frac{n}{2} \right\rfloor, & s = 1, t \geq 3, 4 \leq n \leq 9 \\ \left\lceil \frac{n-1}{4} \right\rceil + 2, & s = 1, t \geq 3, n \geq 10 \\ \left\lceil \frac{n-1}{4} \right\rceil + 1, & s \geq 2, t \geq 3, n \geq 4 \end{cases}$$

Misalkan $V(K_{s,t}) = \{u_1, ..., u_s, v_1, ..., v_t\}$ *dengan partisi pertama adalah* $u_1, ...u_s$ *dan partisi kedua adalah* $v_1, ..., v_t$*. Misalkan pula* $V(C_n) = \{i|0 \leq i \leq n-1\}$ *dengan* $0, 1, ..., n-1, 0$ *lingkaran terbesar di* C_n*.*

Untuk $s = 1, t \geq 3, 4 \leq n \leq 9$*, perhatikan bahwa salah salah satu pasang titik dengan jarak terpanjang adalah titik* $(0, t_1)$ *dan titik* $(\left\lfloor \frac{n}{2} \right\rfloor, t_2)$*, dengan jaraknya ialah* $\left\lfloor \frac{n}{2} \right\rfloor$*.*

Untuk kasus kedua dan ketiga, kita dapat memilih lintasan untuk jarak terpanjang di graf lingkaran 2-kuat graf bipartit lengkap dengan cara yang mirip dilakukan pada Observasi 3 pada subgraf yang isomorfis dengan $P_3 \boxtimes_2 P_{k+1}$*, dengan* $k = \left\lfloor \frac{n}{2} \right\rfloor$*.*

Berdasarkan Teorema 4 dan Observasi 5, diperoleh hasil untuk graf hasil kali 2-kuat berdiameter 2 berikut.

Akibat 12.3.

$$4 \le pd(C_4 \boxtimes_2 K_{s,t}) \le 4(s+t) - 1 \qquad s \ge 1, t \ge 3$$
$$4 \le pd(C_5 \boxtimes_2 K_{s,t}) \le 5(s+t) - 1 \qquad s \ge 1, t \ge 3$$

3.3.4 Dimensi Partisi Graf Bipartit Lengkap 2-Kuat Graf Bipartit Lengkap

Observasi 6.

$$diam(K_{m,n} \boxtimes_2 K_{s,t}) = 2, \qquad m, s \ge 1, n, t \ge 3$$

Graf $K_{q,r}$ dan $K_{s,t}$ masing-masing memiliki diameter 2, sehingga $diam(K_{m,n} \boxtimes_2 K_{s,t}) \le 2$. Namun karena graf hasil kali 2-kuat tersebut bukanlah graf lengkap, maka $diam(K_{q,r} \boxtimes_2 K_{s,t}) = 2$.

Berdasarkan Teorema 4 dan Observasi 6, diperoleh hasil sebagai berikut.

Akibat 12.4. *Misalkan $pd(K_{q,r} \boxtimes_2 K_{s,t}) = k$ maka $p + q + r + s \le k2^k$.*

Akibat 12.5.

$$4 \le pd(K_{m,n} \boxtimes_2 K_{s,t}) \le (n+m)(s+t) - 1 \qquad m, s \ge 1, n, t \ge 3$$

3.3.5 Karakterisasi Dimensi Partisi 3 untuk Graf Hasil Kali 2-Kuat

Melihat sejauhi ini, untuk graf berorde kecil, kita hanya mendapatkan graf berdimensi partisi 3 adalah $P_3 \boxtimes_2 P_3$ dengan kemungkinan 3-partisi yang sangat sedikit, maka dibuatlang konjektur berikut.

Konjektur 1.

$$pd(G \boxtimes_2 H) = 3 \Leftrightarrow G = H = P_3$$

Pernyataan dari ruas kanan ke ruas kiri telah ditunjukkan oleh Proposisi 2. Namun pernyataan arah sebaliknya belum bisa dibuktikan.

Bab 4 Program Pencari Dimensi Partisi Graf-Graf Tertentu

4.1 Perancangan Arsitektur Program

Program dibuat dengan bahasa pemrograman Python versi 3.6. Dibuat dua jenis program, yaitu program yang memiliki antar muka umum (*general user interface*/GUI) dan program dalam bentuk *notebook*.

Pada program GUI, program dibagi menjadi tiga skrip Python yang berbeda:

1. *main.py*, skrip yang dieksekusi
2. *partition.py*, skrip yang menyimpan fungsi pencari partisi suatu himpunan
3. *graph_object.py*, skrip untuk menyimpan kelas *Graph* yang dibuat

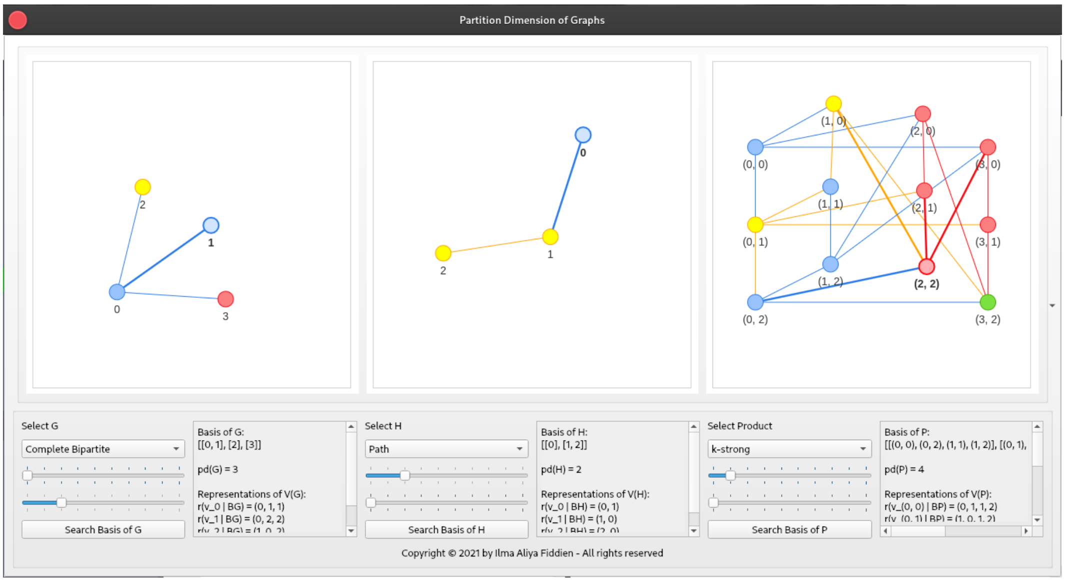
Sedangkan pada program *notebook*, program hanya menggunakan satu berkas IPython Notebook dengan nama *graph-partition-dimension.ipynb*.

4.2 Antarmuka dan Fitur Program

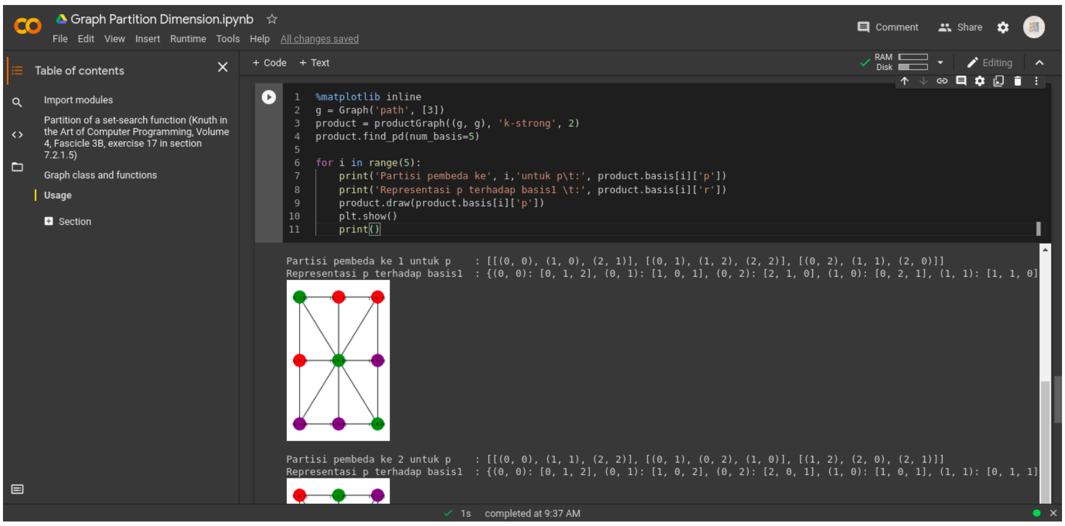
Antarmuka program GUI dibangun menggunakan modul PyQt5. Terdapat dua bagian utama antarmuka, yaitu visualisasi graf dan hasil pencarian partisi pembeda, serta tombol pemilihan graf dan parameternya.

Pengguna dapat memilih dua jenis graf dengan cara memilih nama jenis graf pada menu *dropdown*. Kemudian pengguna dapat mengatur orde graf dengan memindahkan *slider* yang ada di bawahnya. Gambar representasi graf akan muncul secara otomatis.

Untuk mencari dimensi partisi, pengguna mengetuk tombol "Search Basis of ..." lalu program akan memulai mengecek semua kemungkinan partisi yang dapat membedakan graf tersebut dimulai dari kardinalitas partisi yang kecil. Jika pencarian dimensi partisi telah selesai, tiap titik pada graf memiliki warna sesuai dengan partisi pembedanya. Selanjutnya jika kursor diarahkan pada suatu titik, akan muncul vektor representasi titik tersebut terhadap partisi yang sedang ditampilkan.



Gambar 4.1: Program GUI



Gambar 4.2: Program Notebook

4.3 Algoritma Program

Algorithm 1: Mencari dimensi partisi dengan *brute-force*

Result: mendapat $pd = pd(G)$

G := $Graph()$;

V := G.graph.nodes();

k := 2;

upper_bound := G.graph.order();

while *k <= upper_bound* **do**

    $partitions \leftarrow all\_partition(V, k)$;

    forall *partition in partitions* **do**

        if *is_resolving(partition, G)* **then**

            $pd \leftarrow length(partition)$;

            return pd;

        else

            next $partition$;

        end

    end

    $k \leftarrow k + 1$;

end

4.4 Kelebihan dan Kekurangan Program

Kelebihan dari program yang telah dibuat adalah sebagai berikut.

1. Terdapat pilihan program dengan format *notebook* yang fleksibel diubah-ubah untuk penelitian.
2. Terdapat pilihan program dengan format *GUI* yang cocok untuk visualisasi graf.
3. Pada program GUI, pengguna dapat berinteraksi langsung dengan graf dengan cara memindah-mindahkan titik pada gambar untuk membantu intuisi bangun ruang.

Kekurangan dari program yang telah dibuat adalah sebagai berikut.

1. Pengguna harus membiasakan diri dengan bahasa pemrograman Python untuk menggunakan program *notebook*.
2. Masih sedikit pilihan jenis graf dan jenis operasi graf yang bisa dipilih pada kedua jenis program.
3. Program mulai berjalan cukup lama dalam mencari dimensi partisi graf yang ordenya melebihi 12.

Bab 5 Penutup

5.1 Kesimpulan

Dari penelitian yang telah dilakukan, didapat kesimpulan sebagai berikut.

1. Graf hasil kali 2-kuat memiliki dimensi partisi paling kecil 3 dan memiliki batas atas perkalian dimensi partisi dari graf kuadrat masing-masing graf komponennya. Belum ditemukan graf yang ketat memenuhi batas atas tersebut.

2. Belum ditemukan satu nilai dimensi partisi graf hasil kali kuat antara kelas graf lintasan, lingkaran, dan bipartit lengkap, kecuali untuk graf-graf yang partikular. Adapun rentang nilainya dimensi partisinya dapat ditentukan dengan memanfaatkan properti diameter. Hasilnya sebagai berikut.

$$
\begin{array}{rl}
\text{(i) } 3 \leq pd(P_m \boxtimes_2 P_n) \leq 9 & m,n \geq 3 \\
\text{(ii) } 4 \leq pd(P_3 \boxtimes_2 K_{s,t}) \leq 3(s+t)-1 & s \geq 1, t \geq 3 \\
\text{(iii) } 4 \leq pd(C_4 \boxtimes_2 K_{s,t}) \leq 4(s+t)-1 & s \geq 1, t \geq 3 \\
\text{(iv) } 4 \leq pd(C_5 \boxtimes_2 K_{s,t}) \leq 5(s+t)-1 & s \geq 1, t \geq 3 \\
\text{(v) } 4 \leq pd(K_{m,n} \boxtimes_2 K_{s,t}) \leq (n+m)(s+t)-1 & m,s \geq 1, n,t \geq 3
\end{array}
$$

3. Program yang dapat membantu mencari partisi pembeda beberapa kelas graf tertentu dapat dibuat menggunakan bahasa pemrograman Python dengan dua pilihan format, yaitu dalam format *IPython notebook* atau dalam format program yang memiliki GUI.

5.2 Saran

Dari pengalaman penelitian yang telah dilakukan dan dari kesimpulan di atas, berikut saran penelitian atau aplikasi lebih lanjut dari karya tulis ini.

1. Dimensi partisi dan dimensi metrik graf kuasa ke-k memiliki potensi untuk

dieksplorasi lebih lanjut.

2. Definisi graf hasil kali kuat dapat diperumum lagi menjadi graf hasil kali (k, l)-kuat. Yaitu syarat sisi kuat menjadi $d_G(u, v) = k$ dan $d_H(x, y) = l$.

3. Program memuat kelas-kelas graf dan operasi graf yang lebih bervariasi dan memuat algoritma pencarian dimensi metrik.

DAFTAR PUSTAKA

Chappell, G. G., Gimbel, J., & Hartman, C. (2008). Bounds on the metric and partition dimensions of a graph. *Ars Combinatoria*, *88*.

Chartrand, G., Salehi, E., & Zhang, P. (2000). The partition dimension of a graph. *Aequationes Mathematicae*, *59*(1-2), 45–54.

Chvátal, V. (1983). Mastermind. *Combinatorica*, *3*(3), 325–329.

Garey, D. S., M. R. dan Johnson. (1979). *Computers and intractability: A guide to the theory of np-completeness*. Freeman.

Harary, F., & Melter, R. A. (1976). On the metric dimension of graph. *Ars Combinatoria*, *2*, 191–195.

Johnson, M. A. (1993). Structure-activity maps for visualizing the graph variables arising in drug design. *Journal of Biopharmaceutical Statistics*, *3*, 203–236.

Khuller, S., Raghavachari, B., & Rosenfeld, A. (1996). Landmarks in graphs. *Discrete Applied Mathematics*, *70*, 217–229.

Melter, R. A., & Tomescu, I. (1984). Metric bases in digital geometry. *Computer Vision Graphics and Image Processing*, *25*, 113–121.

Puš, V. (1991). Combinatorial properties of products of graphs. *Czechoslovak Mathematical Journal*, *41*(2), 269–277. https://doi.org/10.21136/CMJ.1991.102459

Slater, P. J. (1975). Leaves of trees. *Congressus Numerantium*, *14*, 549–559.

Spinelli, B., Celis, E., & Thiran, P. (2017). A general framework for sensor placement in source localization. *IEEE Transactions on Network Science and Engineering*, *2*, 86–102.

Tomescu, I. (2008). Discrepancies between metric dimension and partition dimension of a connected graph. *Discrete Mathematics*, *308*(22), 5026–5031.

Yero, I. G., Jakovac, M., Kuziak, D., & Taranenko, A. (2014). The partition dimension of strong product graphs and cartesian product graphs. *Discrete Mathematics*, *331*, 43–52.

LAMPIRAN

Kode *Script* Python Untuk Mengenerasi Semua Partisi dari Suatu Himpunan

```
def partition(Set, m):
    """ a function to create all possible partitions of a set """

    def visit(n, a):
        ps = [[] for i in range(m)]
        for j in range(n):
            ps[a[j + 1]].append(Set[j])
        return ps

    def f(mu, nu, sigma, n, a):
        if mu == 2:
            yield visit(n, a)
        else:
            for v in f(mu - 1, nu - 1, (mu + sigma) % 2, n, a):
                yield v
        if nu == mu + 1:
            a[mu] = mu - 1
            yield visit(n, a)
            while a[nu] > 0:
                a[nu] -= 1
                yield visit(n, a)
        elif nu > mu + 1:
            if (mu + sigma) % 2 == 1:
                a[nu - 1] = mu - 1
            else:
                a[mu] = mu - 1
            if (a[nu] + sigma) % 2 == 1:
                for v in b(mu, nu - 1, 0, n, a):
                    yield v
            else:
                for v in f(mu, nu - 1, 0, n, a):
                    yield v
```

```
            while a[nu] > 0:
                a[nu] -= 1
                if (a[nu] + sigma) % 2 == 1:
                    for v in b(mu, nu - 1, 0, n, a):
                        yield v
                else:
                    for v in f(mu, nu - 1, 0, n, a):
                        yield v

def b(mu, nu, sigma, n, a):
    if nu == mu + 1:
        while a[nu] < mu - 1:
            yield visit(n, a)
            a[nu] += 1
        yield visit(n, a)
        a[mu] = 0
    elif nu > mu + 1:
        if (a[nu] + sigma) % 2 == 1:
            for v in f(mu, nu - 1, 0, n, a):
                yield v
        else:
            for v in b(mu, nu - 1, 0, n, a):
                yield v
        while a[nu] < mu - 1:
            a[nu] += 1
            if (a[nu] + sigma) % 2 == 1:
                for v in f(mu, nu - 1, 0, n, a):
                    yield v
            else:
                for v in b(mu, nu - 1, 0, n, a):
                    yield v
        if (mu + sigma) % 2 == 1:
            a[nu - 1] = 0
        else:
            a[mu] = 0
    if mu == 2:
        yield visit(n, a)
```

```
        else:
            for v in b(mu - 1, nu - 1, (mu + sigma) % 2, n, a):
                yield v

    n = len(Set)
    a = [0] * (n + 1)
    for j in range(1, m + 1):
        a[n - m + j] = j - 1
    return f(m, n, 0, n, a)
```

Berkas program dapat dilihat dan diunduh di link berikut. *https://github.com/ilmaaliyaf/graph-partition-dimension*

Kode *Script* Python Untuk Membuat Objek Graf & Pencari Partisi Dimensi

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Apr 15 10:13:31 2021
@author: ilmaaliyaf
"""
import networkx as nx
from partition import partition

class Graph(nx.graph.Graph):

    def __init__(self, graph_type='', parameter=[]):
        if graph_type not in ['path', 'cycle', 'complete', 'complete_bipartite']:
            self.graph = nx.empty_graph()
            self.graph_params = None
        else:
            self.define_type(graph_type, parameter)
        self.basis = [{'p': '', 'r': ''}]
        self.pd = None
        self.count = 0

    def define_type(self, graph_type, parameter):
        self.graph_type = graph_type
        self.graph_params = parameter
        if graph_type not in ['path', 'cycle', 'complete', 'complete_bipartite']:
            raise ("Choose type between path, cycle, complete, and complete_bipartite")
        else:
            self.graph = eval('nx.' + graph_type + '_graph(*parameter)')
```

```python

    def diam(self):
        return nx.diameter(self.graph)

    def distance(self):
        return dict(nx.all_pairs_shortest_path_length(self.graph))

    def find_pd(self,
                num_basis=1,
                lower_bound=2,
                upper_bound='',
                print_result=False):
        """ find basis for this graph """
        V = list(self.graph)
        basis_ = []
        if type(upper_bound) != int:
            upper_bound = self.graph.order() + 1

        i = 0
        for k in range(lower_bound, upper_bound):
            partitions = partition(V, k)
            for j, P in enumerate(partitions):
                if j < self.count:
                    continue
                resolving, representation = self.is_resolving(P)
                if resolving:  # add P into basis_dict
                    basis_.append({'p': P, 'r': representation})
                    self.basis = basis_
                    self.pd = len(basis_[-1]['p'])
                    i += 1
                    if print_result:
                        print(P)
                    if i + 1 > num_basis:
                        self.count = j
                        return

    def is_resolving(self, partition):
```

```
        r = {}
        d = self.distance()
        for v in self.graph.nodes:
            r[v] = []
            for P in partition:
                dvP = d[v][min(P, key=d[v].get)]
                r[v].append(dvP)
        r_reversed = {str(val): key for key, val in r.items()}
        return len(r) == len(r_reversed), r

    def r(self, partition):
        r = {}
        d = self.distance()
        for v in self.graph.nodes:
            r[v] = []
            for P in partition:
                dvP = d[v][min(P, key=d[v].get)]
                r[v].append(dvP)
        return r

class productGraph(Graph):

    def __init__(self, G, H, product_type, product_params):
        self.component_graphs = (G, H)
        self.product_type = product_type
        self.product_params = product_params
        self.graph_type = 'product'

        if product_type == 'k-strong':
            self.k_strong()
        elif product_type == 'cartesian':
            self.cartesian()
        else:
            self.graph = nx.empty_graph()

        self.count = None
        self.basis = [{'p': '', 'r': ''}]
```

```
        self.pd = None

    def k_strong(self):
        G = self.component_graphs[0]
        H = self.component_graphs[1]
        if self.product_params == 1:
            self.graph = nx.strong_product(G.graph, H.graph)
        else:
            P = nx.cartesian_product(G.graph, H.graph)
            dG = G.distance()
            dH = H.distance()
            jumping_vertex = []
            for u in P.nodes:
                for v in P.nodes:
                    if dG[u[0]][v[0]] == self.product_params \
                            and dH[u[1]][v[1]] == self.product_params:
                        P.add_edge(u, v)
                        if v not in jumping_vertex:
                            jumping_vertex.append(v)
            self.graph = P
            self.jumping_vertex = jumping_vertex

    def cartesian(self):
        G = self.component_graphs[0]
        H = self.component_graphs[1]
        self.graph = nx.cartesian_product(G.graph, H.graph)
```

Kode *Script* Python Untuk Membuat Program GUI

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Apr 15 10:13:31 2021
@author: ilmaaliyaf
"""
import sys
import PyQt5.QtWidgets as Widget
from PyQt5.QtCore import Qt
from PyQt5.QtWebEngineWidgets import QWebEngineView
import networkx as nx
from graph_object import Graph, productGraph
from pyvis.network import Network

class MainProgram(Widget.QComboBox):

    def __init__(self):
        super(MainProgram, self).__init__()
        self.initials = ('g', 'h', 'p')
        self.graph_type = ['Path', 'Cycle', 'Complete', 'Complete Bipartite']
        self.product_type = ['Cartesian', 'k-strong']
        self.initial_graph_param = {'path': 2, 'cycle': 3, 'complete': 2,
                                    'complete_bipartite': 1, 'k-strong':1,
                                    '': 1, 'cartesian': 1}
        # === LEFT SECTION INITIALISATION: SLIDERS ===
        def init_slider_widget():
            slider = Widget.QSlider(Qt.Horizontal)
            slider.setValue(1)
            slider.setFocusPolicy(Qt.StrongFocus)
            slider.setTickPosition(Widget.QSlider.TicksBothSides)
            slider.setTickInterval(1)
```

```
            slider.setSingleStep(1)
            slider.setMinimum(1)
            slider.setMaximum(10)
            slider.setSingleStep(1)
            return slider
        self.slider = {}
        self.view = {}
        for x in self.initials:
            self.slider[x] = (init_slider_widget(), init_slider_widget())
            self.view[x] = QWebEngineView()
        # === GRAPH OBJECTS INITIALISATION ===
        self.select_graph = {}
        for x in self.initials[:2]:
            self.select_graph[x] = Widget.QComboBox(self)
            self.select_graph[x].addItems(self.graph_type)
            self.select_graph[x].setCurrentIndex(-1)
        self.select_graph['p'] = Widget.QComboBox(self)
        self.select_graph['p'].addItems(self.product_type)
        self.select_graph['p'].setCurrentIndex(-1)
        # === BASIS BUTTON INITIALISATION ===
        self.basisButtonBox = Widget.QGroupBox()
        self.result_button = {}
        for i, x in enumerate(self.initials):
            self.result_button[x] = Widget.QPushButton()
            self.result_button[x].setText(f'Search Basis of {x.upper()}')
            self.result_button[x].clicked.connect(self.basis_updater)
            self.result_button[x].setEnabled(False)
        # === RESULT BOX INITIALISATION ===
        self.viewBox = Widget.QGroupBox()
        self.resultBox = Widget.QGroupBox()
        self.result = {}
        for x in self.initials:
            self.result[x] = Widget.QLabel(
                f'Basis of {x.upper()}:\nNone\nRepresentations of
```

```
                    V({x.upper()}):\nNone', self)
        gr = Graph('path', [1])
        # === GRAPH-RELATED VARIABLES INITIALISATION ===
        self.w = '409px'
        self.h = '420px'
        self.nt = {x: {'type': '', 'params': '', 'pd': 100,
                       'nt': Network(height=self.h, width=self.w),
                       'nx': gr}
                   for x in self.initials}
        # CALLING UI
        self.unit_ui()
        self.show()
        self.setWindowState(Qt.WindowMaximized)

    def center(self):
        qr = self.frameGeometry()
        cp = Widget.QDesktopWidget().availableGeometry().center()
        qr.moveCenter(cp)
        self.move(qr.topLeft())

    def unit_ui(self):
        self.setGeometry(100, 100, 1200, 600)
        self.center()
        self.setWindowTitle('Partition Dimension of Graphs')
        grid = Widget.QGridLayout()
        grid.setColumnStretch(1, 6)
        grid.setRowStretch(1, 2)
        self.setLayout(grid)
        # ADD SELECT GRAPH & RESULT SECTION
        self.box_result()
        result_layout = Widget.QHBoxLayout()
        result_layout.addWidget(self.resultBox)
        grid.addLayout(result_layout, 2, 0, 1, 3)
        # ADD GRAPH VIEWER
        self.viewer()
        view_layout = Widget.QHBoxLayout()
        view_layout.addWidget(self.viewBox)
```

```
        grid.addLayout(view_layout, 1, 1),

    def viewer(self):
        layout = Widget.QGridLayout()

        for i, x in enumerate(self.initials):
            self.nt[x]['nt'].from_nx(self.nt[x]['nx'].graph)
            self.nt[x]['nt'].save_graph(f'view_graph_{x}.html')
            with open(f'view_graph_{x}.html', 'r') as f:
                html = f.read()
            self.view[x].setHtml(html)
            layout.addWidget(self.view[x], 0, i, alignment=Qt.
                AlignCenter)

        layout.setSpacing(10)
        self.viewBox.setLayout(layout)

    def box_result(self):
        layout = Widget.QGridLayout()
        layout.setSpacing(10)

        label = {}
        label['g'] = Widget.QLabel('Select G', self)
        label['h'] = Widget.QLabel('Select H', self)
        label['p'] = Widget.QLabel('Select Product', self)

        for i, x in enumerate(self.initials):
            layout.addWidget(label[x], 0, i*2)
            layout.addWidget(self.select_graph[x], 1, i*2)
            layout.addWidget(self.result_button[x], 4, i*2)
            scroll = Widget.QScrollArea()
            scroll.setWidget(self.result[x])
            scroll.setWidgetResizable(True)
            scroll.setFixedHeight(150)
            scroll.setViewportMargins(5, 5, 5, 5)
            layout.addWidget(scroll, 0, i*2+1, 5, 1)
            # manage signal
```

```
            self.select_graph[x].activated.connect(self.update_graph)
            self.select_graph[x].activated[str].connect(self.on_product)
            self.select_graph[x].currentIndexChanged['QString'].connect(self.disable_widget)

            # GRAPH SLIDERS
            for j in range(2):
                # add to layout
                layout.addWidget(self.slider[x][j], j+2, i*2)
                # manage signal
                self.slider[x][j].valueChanged.connect(self.update_graph)
                self.slider[x][j].valueChanged[int].connect(self.on_product)

        self.select_graph['p'].setEnabled(False)
        self.slider['p'][0].setEnabled(False)
        self.slider['p'][1].setEnabled(False)

        notice = Widget.QLabel('Copyright 2021 by Ilma Aliya Fiddien - All rights reserved', self)
        layout.addWidget(notice, 5, 0, 6, 0, Qt.AlignCenter)

        self.resultBox.setLayout(layout)

    def disable_widget(self, currentIndex):
        # if both g and h are active, then make p active
        if self.select_graph['h'].isEnabled() and self.select_graph['h'].isEnabled():
            self.select_graph['p'].setEnabled(True)
        else:
            self.select_graph['p'].setEnabled(False)

        # if p is active then make its sliders and button active
        if self.select_graph['p'].isEnabled() == False:
```

```
            self.slider['p'][0].setEnabled(False)
            self.slider['p'][1].setEnabled(False)
            self.result_button['p'].setEnabled(False)

        sender = self.sender()
        # if g or h are in those list, make the 2nd slider (parameter) active
        if currentIndex in ['Complete Bipartite']:
            if sender is self.select_graph['g']:
                self.slider['g'][1].setEnabled(True)
            if sender is self.select_graph['h']:
                self.slider['h'][1].setEnabled(True)
        else:
            if sender is self.select_graph['g']:
                self.slider['g'][1].setEnabled(False)
            if sender is self.select_graph['h']:
                self.slider['h'][1].setEnabled(False)

        if currentIndex in ['k-strong']:
            self.slider['p'][0].setEnabled(True)
        elif currentIndex in ['cartesian']:
            self.slider['p'][0].setEnabled(False)
            self.slider['p'][1].setEnabled(False)
        else:
            self.slider['p'][0].setEnabled(False)

    # ============================
    # DRAWING ON THE CANVAS
    # ============================

    def graph_drawer(self, graph, title, x):
        """ Function to draw the updated graph into it's canvas """
        graph = nx.relabel_nodes(graph, lambda node: str(node))
        nt = Network(height=self.h, width=self.w)
        nt.from_nx(graph)

```

```
        if x == 'p':
            layout = {}
            for v in graph.nodes:
                v_ = eval(v)
                layout[v] = ([v_[0], v_[1]])
            for node in nt.nodes:
                node["x"] = layout[node['id']][0] * 100
                node["y"] = layout[node['id']][1] * 100
            nt.toggle_physics(False)

        nt.save_graph(f'view_graph_{x}.html')
        with open(f'view_graph_{x}.html', 'r') as f:
            html = f.read()
            self.view[x].setHtml(html)
        self.nt[x]['nt'] = nt

    def update_graph(self):
        """ Function to set graph G """
        s = self.sender()
        if s in [self.select_graph['g'], self.slider['g'][0], self.slider['g'][1]]:
            x = 'g'
        elif s in [self.select_graph['h'], self.slider['h'][0], self.slider['h'][1]]:
            x = 'h'
        else:
            x = 'p'

        graph_type = '_'.join(str(self.select_graph[x].currentText()).lower().split())
        m = self.slider[x][0].value()
        if m < self.initial_graph_param[graph_type]:
            m = self.initial_graph_param[graph_type]
            self.slider[x][0].setValue(self.initial_graph_param[graph_type])
        if self.slider[x][1].isEnabled():
            n = self.slider[x][1].value()
```

```
            graph = Graph(graph_type, [m, n])
            graph_params = [m, n]
        else:
            graph = Graph(graph_type, [m])
            graph_params = [m]
        self.nt[x]['nx'] = graph
        self.graph_drawer(graph.graph, f'{graph_type} {
            graph_params}', x)

        # enabling basis updater button
        self.result_button[x].setEnabled(True)
        self.result[x].setText(f'Basis of {x.upper()}:\nNone\
            nRepresentations of V({x.upper()}):\nNone')
        self.change = True

    def on_product(self):
        """ Function to set graph product P """
        product_type = '_'.join(str(self.select_graph['p'].
            currentText()).lower().split())
        p = productGraph(self.nt['g']['nx'], self.nt['h']['nx'],
                              product_type, self.slider['p'][0].
                                  value())
        self.nt['p']['nx'] = p
        self.graph_drawer(p.graph, p.product_type + " " + str(p.
            product_params), 'p')

        # enabling basis updater button
        self.result_button['p'].setEnabled(True)
        self.change = True

    # =====================
    # DEALING WITH THE BASIS
    # =====================

    def basis_updater(self):
        """ Function to update basis of the updated graph """
        # search partition dimension and it's basis
```

```python
        s = self.sender()
        if s == self.result_button['g']:
            x = 'g'
        elif s == self.result_button['h']:
            x = 'h'
        else:
            x = 'p'
        # prepare to continue checking the partitions from last
            iteration
        # because we are dealing with the same graph
        if self.change == False:
            count = self.nt[x]['nx'].count
            num_basis = 2
            pd = self.nt[x]['pd']
        else:
            count = 0
            num_basis = 1
            pd = self.nt[x]['nx'].graph.order()

        graph = self.nt[x]['nx']
        graph.count = count # continue from the last checked
            partition
        graph.find_pd(num_basis=num_basis) # search for partition
            dimension
        basis = graph.basis[-1]['p']
        r = graph.basis[-1]['r']

        for v in graph.graph.nodes:
            graph.graph.nodes[v]['title'] = str(r[v])
            graph.graph.nodes[v]['group'] = r[v].index(0) + 1

        self.graph_drawer(graph.graph, graph.graph_type, x)

        # 2 form the text to show
        G = x.upper()

        if pd < graph.pd:
```

```
            part_type = 'Resolving partition'
            connection = '<='
        else:
            part_type = 'Basis'
            connection = '='

        b_text = part_type + ' of ' + G + ':\n' \
                + str(basis) \
                + '\n\npd(' + G + ') '+ connection + ' ' + str(
                    graph.pd)
        rep = ''
        for v in r.keys():
            rep = rep + 'r(v_' + str(v) + ' | B' + G + ') = ' \
                    + str(tuple(r[v])) + '\n'

        b_text = b_text + '\n\nRepresentations of V(' + G \
                        + '):\n' + rep

        # 3 send signal to result box
        self.result[x].setText(b_text)

        self.change = False

if __name__ == '__main__':
    app = Widget.QApplication(sys.argv)
    app.setStyle("Fusion")
    screen = MainProgram()
    result = app.exec_()
    del screen
    del app
    sys.exit(result)
```