MATEMATIKA DISKRIT

Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

YAYASAN PRIMA AGUS TEKNIK

MATEMATIKA DISKRIT

Penulis :
Dr. Ir. Agus Wibowo, M.Kom., M.Si., MM.

ISBN : 9 786235 734781

Editor :
Dr. Joseph Teguh Santoso, S.Kom., M.Kom.

Penyunting :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom.

Penebit :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas STEKOM)

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : info@stekom.ac.id

KATA PENGANTAR

Puji syukur kami panjatkan atas terselesaikannya buku berjudul ***"Matematika Diskrit"*** dengan baik. Matematika Diskrit adalah cabang matematika yang melibatkan unsur-unsur diskrit yang menggunakan aljabar dan aritmatika. Hal ini semakin banyak diterapkan di bidang praktis matematika dan ilmu komputer. Ini adalah alat yang sangat baik untuk meningkatkan penalaran dan kemampuan pemecahan masalah. Buku ini menjelaskan konsep dasar Himpunan, Hubungan dan Fungsi, Logika Matematika, Teori Grup, Teori Hitung, Probabilitas, Induksi Matematika dan Hubungan Perulangan, Teori Grafik, Pohon dan Aljabar Boolean.

Semarang, Agustus 2022
Penulis

Dr. Agus Wibowo, M.Kom, M.Si, MM.

DAFTAR ISI

BAB 1
SET TEORI DAN LOGIKA

1.1 PENGANTAR TEORI HIMPUNAN

Konsep himpunan memainkan peran yang sangat penting dalam semua cabang matematika modern. Dalam beberapa tahun terakhir teori himpunan telah menjadi bidang investigasi yang penting karena caranya meresapi begitu banyak pemikiran matematika kontemporer. Pemahaman sejati dari setiap cabang matematika modern membutuhkan pengetahuan tentang teori himpunan karena itu adalah dasar umum dari berbagai bidang matematika. Set digunakan untuk mengelompokkan objek yang berbeda menjadi satu. Objek-objek yang termasuk dalam himpunan perlu didefinisikan dengan baik dalam arti bahwa tidak boleh ada ambiguitas dalam memutuskan apakah suatu objek tertentu termasuk dalam suatu himpunan atau tidak. Jadi, diberikan suatu objek, baik itu milik set tertentu atau bukan miliknya. Misalnya, lima huruf pertama dari alfabet Inggris merupakan himpunan yang dapat direpresentasikan secara simbolis sebagai himpunan {a, b, c, d, e}. Objek arbitrer termasuk dalam himpunan ini jika dan hanya jika itu adalah salah satu dari lima huruf ini. Kelima objek berbeda ini dapat muncul dalam urutan apa pun dalam representasi ini. Dengan kata lain, himpunan ini juga dapat direpresentasikan dengan {d, b, a, e, c}. Objek yang termasuk dalam himpunan tidak perlu memiliki properti umum. Dengan demikian angka 4, huruf x, dan kata “buku” dapat merupakan himpunan S yang dapat direpresentasikan sebagai S = {*x*, buku, 4}. Suatu hari tertentu mungkin dingin untuk satu orang dan tidak dingin untuk orang lain, jadi "kumpulan hari-hari dingin dalam sebulan" bukanlah kumpulan yang jelas. Demikian pula, “kumpulan bilangan besar” dan “kumpulan laki-laki tinggi” juga bukan himpunan.

Istilah objek telah digunakan di sini tanpa menentukan dengan tepat apa objek itu. Dari sudut pandang matematika, himpunan adalah istilah teknis yang mengambil maknanya dari sifat-sifat yang kita asumsikan dimiliki oleh himpunan. Deskripsi informal dari suatu himpunan, berdasarkan gagasan intuitif dari suatu objek, pertama kali diberikan oleh ahli matematika Jerman Georg Cantor (1845–1918) menjelang akhir abad kesembilan belas dan teori himpunan berdasarkan versinya dikenal sebagai naif. teori himpunan. Dalam kata-kata Cantor sendiri, "satu set menyatukan menjadi keseluruhan objek persepsi kita yang terdefinisi dengan baik dan objek-objek ini adalah elemen dari set." Kumpulan yang dibahas dalam buku ini semuanya dapat dilihat dalam kerangka teori Cantor ini.

Jadi himpunan adalah kumpulan dari objek-objek yang berbeda. Objek-objek dalam himpunan disebut elemen atau anggota himpunan. Jika x adalah anggota himpunan A, kita katakan bahwa x milik A, dan dinyatakan secara simbolis sebagai $x \in A$. Notasi $y \in A$ tersebut menyatakan bahwa y bukan anggota himpunan A.

Himpunan Hingga dan Tak Terbatas

Suatu himpunan berhingga jika jumlah elemen di dalamnya berhingga. Jika tidak, itu adalah himpunan tak terbatas. Himpunan bilangan bulat positif yang kurang dari 100 adalah himpunan berhingga, sedangkan himpunan semua bilangan bulat positif adalah himpunan tak hingga. Jika X adalah himpunan berhingga, kardinalitas dari X adalah banyaknya anggota X, dan bilangan bulat tak negatif ini dilambangkan dengan N(X). Himpunan kardinalitas 1 disebut himpunan tunggal.

Jika suatu himpunan berhingga dan jika kardinalitasnya tidak terlalu besar, kita dapat mendeskripsikannya dengan enumerasi semua elemennya. Misalnya, representasi S = {a, e, i, o, u} menggambarkan himpunan semua vokal alfabet Inggris. Jika kardinalitas terlalu besar, metode enumeratif ini sangat tidak nyaman. Dalam beberapa kasus, jika tidak ada ambiguitas, kami membuat deskripsi enumeratif ini lebih ringkas. Misalnya, himpunan D bilangan bulat positif antara 25 dan 123 dapat direpresentasikan sebagai D = {25, 26, 27, ⋯, 121, 122, 123}. Cara yang lebih baik untuk merepresentasikan D adalah dengan menyatakan properti untuk keanggotaannya. Sebuah objek n adalah elemen dari himpunan D ini jika dan hanya jika n adalah bilangan bulat positif yang paling sedikit 25 dan paling banyak 123. Dengan kata lain, kita menulis:

$$D = \{n : n \text{ adalah bilangan bulat positif}, 24 < n < 124\}$$

Himpunan tak hingga N dari semua bilangan asli dapat dinyatakan dengan jelas sebagai N = {1, 2, 3, . . } atau sebagai N = {n : n adalah bilangan asli} dengan menyatakan kriteria keanggotaannya. Notasi yang merepresentasikan suatu himpunan dengan menyatakan kriteria keanggotaannya seperti dijelaskan di atas disebut dengan notasi pembentuk himpunan.

Subset dari Himpunan dan Himpunan Kosong

Suatu himpunan P adalah himpunan bagian dari himpunan Q jika setiap elemen dari P adalah anggota dari Q. Kita menggunakan notasi $P \subset Q$ untuk menyatakan bahwa P adalah himpunan bagian dari Q. Suatu himpunan bagian dari suatu himpunan tidak diragukan lagi merupakan suatu himpunan bagian. Ketika P adalah himpunan bagian dari Q, kita dapat mengatakan bahwa Q berisi P dan bahwa P terdapat dalam Q. Menurut definisi kita, setiap himpunan adalah himpunan bagian dari dirinya sendiri. Himpunan P adalah himpunan bagian sejati dari Q jika (1) P adalah himpunan bagian dari Q dan (2) paling sedikit ada satu anggota Q yang bukan merupakan anggota P. Himpunan bilangan bulat positif adalah himpunan bagian sejati dari himpunan semua bilangan real.

Jika A adalah himpunan bagian dari B, maka komplemen relatif dari A di B adalah himpunan dari elemen-elemen di B yang bukan merupakan elemen dari A. Komplemen relatif A di B dilambangkan dengan B – A dan dapat digambarkan dengan keanggotaannya kriteria sebagai:

$$B - A = \{t : t \in B, t \notin A\}$$

Dua himpunan saling lepas jika tidak memiliki elemen yang sama. Di sisi lain, dua himpunan adalah sama jika mereka memiliki elemen yang sama. Kita tulis X = Y jika himpunan X dan Y sama. Jelas, dua himpunan adalah sama jika dan hanya jika masing-masing himpunan bagian dari himpunan lainnya. Misalnya, jika X = {r : r adalah akar dari $x^2 - 5x + 6 = 0$} dan Y = {2, 3}, maka X = Y.

Himpunan kosong jika tidak memiliki elemen. Sebuah fakta muncul yang menurut sebagian orang mengejutkan: hanya ada satu set kosong. (Misalkan E dan F adalah dua himpunan kosong. Jika tidak sama, mereka tidak sama. Jadi salah satu dari mereka harus memiliki setidaknya satu elemen yang bukan milik yang lain. Jadi salah satu dari dua set tidak kosong Ini bertentangan dengan asumsi bahwa E dan F keduanya kosong.) Himpunan kosong yang unik (atau himpunan nol) dilambangkan dengan Φ. Fakta bahwa himpunan kosong adalah himpunan bagian dari sembarang himpunan ditetapkan dengan "penalaran hampa": Jika itu bukan himpunan bagian dari himpunan S yang diberikan, harus ada setidaknya satu elemen dalam himpunan kosong yang tidak ada di S. Dalam tertentu, harus ada setidaknya satu elemen dalam himpunan kosong yang merupakan kontradiksi. Tentu saja, suatu himpunan kosong jika dan hanya jika kardinalitasnya nol.

Dalam beberapa kasus kita akan mempertimbangkan himpunan yang semua himpunan bagian dari himpunan U yang disebut himpunan universal. Misalnya, jika himpunan yang dipertimbangkan adalah A, B, dan C, di mana A = {3, 8, 6, 7, x}, B = {8, 4, y, t, 5}, dan C = {3 , 4, x, t, 9}, maka sembarang himpunan yang memuat himpunan D = {3, 8, 6, 7, x, 4, y, t, 5, 9} dapat dianggap sebagai himpunan semesta sejauh A , B, C, dan D yang bersangkutan. Setelah himpunan universal U tetap, komplemen relatif dari subset A di U disebut absolut komplemen dari A dan dilambangkan dengan Ac. Jadi jika alam semesta adalah himpunan semua bilangan bulat nonnegatif dan E adalah himpunan semua genap bilangan, maka Ec adalah himpunan semua bilangan ganjil. Amati bahwa komplemen absolut dari komplemen absolut dari sembarang himpunan A adalah himpunan A itu sendiri.

Set Kekuatan dari Satu Set

Suatu himpunan dapat memiliki himpunan lain sebagai elemen-elemennya. Misalnya himpunan S yang terdiri dari huruf x, himpunan {a, b} dan bilangan 4 dilambangkan dengan S = {x, {a, b}, 4}. Himpunan himpunan bagian juga dikenal sebagai kelas atau keluarga himpunan. Kelas dari semua himpunan bagian dari himpunan X disebut himpunan pangkat X dan dilambangkan dengan P(X). Misalnya, jika X = {1, 2}, anggota P(X) adalah himpunan kosong, himpunan tunggal {1}, himpunan tunggal {2}, dan himpunan X. Jadi P(X) = {Φ , {1}, {2}, {1, 2}}.

Produk Cartesian dari Set

N-tupel terurut (a_1, a_2, a_3, . . , an) adalah kumpulan dari n objek a_1, a_2, . . . , an di mana a1 adalah elemen pertama, a_2 adalah elemen kedua, . . . , dan an adalah elemen ke-n. Dalam n-tupel yang dipesan, elemen-elemennya tidak perlu berbeda. Suatu himpunan dengan n elemen dengan demikian merupakan n-tupel tak beraturan dari n elemen berbeda, karena dalam suatu himpunan urutan di mana elemen-elemen tersebut dianggap tidak relevan. 2-tupel terurut disebut pasangan terurut. Dua n-tupel terurut (a_1, a_2, . . . , a_n) dan (b_1, b_2, . . , b_n) dikatakan sama jika ai = bi untuk i = 1, 2, . . . , n. Himpunan semua pasangan terurut (a, b), di mana a adalah elemen dari himpunan A dan b adalah elemen dari himpunan B, disebut produk kartesius dari A dan B dan dilambangkan dengan A × B. Di lain kata-kata:

$$A \times B = \{(a, b) : a \in A \text{ dan } b \in B\}$$

Misalnya, jika A = {1, 2} dan B = {1, 3}, maka hasil kali kartesius A × B adalah himpunan {(1, 1), (1, 3), (2, 1), (2, 3)}. Lebih umum, produk kartesius dari himpunan A_1, A_2, . . . , A_n dilambangkan dengan $A_1 \times A_2 \times \cdots \times A_n$ adalah himpunan semua n-tupel terurut dari bentuk $(a_1, a_2, . . , a_n)$, di mana ai adalah sembarang elemen dari Ai (i = 1, 2 , . . . , n).

Persimpangan dan Persatuan Himpunan

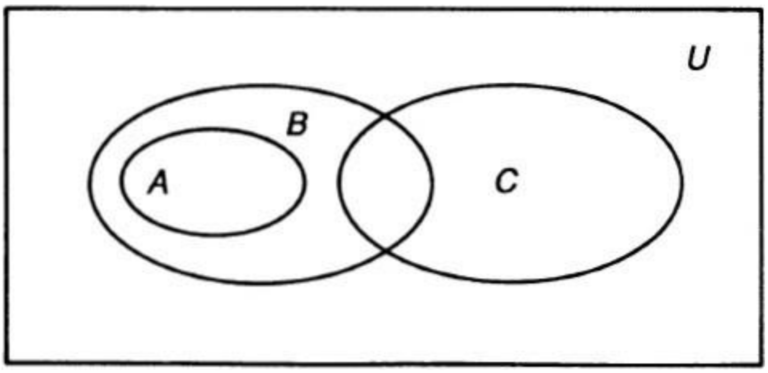
Ada dua konstruksi penting yang dapat diterapkan pada himpunan bagian dari himpunan untuk menghasilkan himpunan bagian baru. Misalkan A dan B adalah dua himpunan bagian dari suatu himpunan X. Himpunan yang terdiri dari semua elemen yang sama untuk A dan B disebut persimpangan A dan B dan dilambangkan dengan $A \cap B$. Jelas, persimpangan suatu himpunan dan himpunan himpunan kosong adalah himpunan kosong dan perpotongan dari sembarang himpunan A dan A adalah A. Juga, perpotongan suatu himpunan dan komplemen mutlaknya adalah kosong karena tidak ada elemen yang dapat berada secara bersamaan di A dan tidak di A. Selain itu, ia mengikuti dari definisi bahwa perpotongan himpunan memiliki sifat komutatif: Perpotongan A dan B sama dengan perpotongan B dan A.

Himpunan yang terdiri dari semua anggota A atau B atau keduanya A dan B disebut gabungan A dan B dan dilambangkan dengan $A \cup B$. Gabungan himpunan A dan himpunan kosong adalah himpunan A dan gabungan dari A dan A juga A. Himpunan himpunan juga komutatif: $A \cup B = B \cup A$. Lebih umum, perpotongan suatu kelas himpunan adalah himpunan elemen (jika ada) yang dimiliki oleh setiap himpunan kelas. Gabungan dari suatu kelas himpunan adalah himpunan dari elemen-elemen yang dimiliki oleh setidaknya satu himpunan dalam kelas tersebut. Ini adalah konsekuensi langsung dari definisi bahwa perpotongan himpunan dan himpunan serikat memiliki sifat asosiatif: (1) $A \cap (B \cap C) = (A \cap B) \cap C$ dan (2) $A \cup (B \cup C) = (A \cup B) \cup C$. Jadi yang pertama dapat ditulis sebagai $A \cap B \cap C$ dan yang terakhir sebagai $A \cup B \cup C$ dengan jelas.

Dua himpunan terputus jika dan hanya jika perpotongannya kosong. Suatu kelas dari himpunan-himpunan adalah disjoint berpasangan jika perpotongan dari dua himpunan di dalam kelas tersebut kosong. Kelas C(X) dari himpunan bagian dari himpunan X disebut partisi dari X jika (1) C(X) adalah disjoint berpasangan, dan (2) gabungan dari himpunan di C(X) adalah himpunan X. Untuk misalnya, kelas {{2, 4}, (1, 3, 5}, {6}} adalah partisi dari himpunan {1, 2, 3, 4, 5, 6}.

Diagram Venn dari Himpunan

Sebuah perangkat yang sangat berguna dan sederhana untuk mewakili set grafis untuk menggambarkan hubungan antara mereka adalah diagram Venn, dinamai ahli logika Inggris John Venn (1834-1923). Dalam diagram Venn, himpunan semesta U yang berisi semua objek yang ditinjau biasanya diwakili oleh persegi panjang, dan di dalam persegi panjang ini himpunan bagian dari himpunan semesta diwakili oleh lingkaran, persegi panjang, atau beberapa bangun geometris lainnya. Dalam diagram Venn yang ditunjukkan pada Gambar 0.1.1, kita memiliki tiga himpunan A, B, dan C yang merupakan himpunan bagian dari himpunan semesta U. Gambar elips yang mewakili himpunan A di dalam elips yang mewakili himpunan B menunjukkan bahwa A adalah himpunan bagian dari B. Fakta bahwa A dan C saling lepas diperjelas dengan merepresentasikannya dengan dua elips yang tidak berpotongan. Fakta bahwa perpotongan B dan C tidak kosong diperjelas dengan menunjukkan bahwa dua elips yang mewakili kedua himpunan ini saling tumpang tindih. Daerah dalam persegi panjang (yang mewakili himpunan semesta) yang berada di luar elips yang mewakili ketiga himpunan tersebut merupakan komplemen mutlak dari gabungan ketiga himpunan tersebut.



Gambar 1.1 Himpunan dalam diagram Venn

Hukum Distributif dan Hukum De Morgan

Kami menyimpulkan bagian ini pada himpunan dengan dua teorema berikut yang terkait dengan operasi himpunan yang melibatkan persimpangan, serikat pekerja, dan pengambilan komplemen absolut. Teorema-teorema ini dapat dengan mudah ditetapkan dengan menggambar diagram Venn. Akan tetapi, sangat bermanfaat untuk membuktikannya tanpa bantuan diagram Venn, karena dalam banyak kasus tidak mungkin untuk merepresentasikan himpunan yang sedang dipertimbangkan dengan diagram tersebut, seperti yang akan kita lihat nanti dalam buku ini.

TEOREMA 1.1 (Hukum Distributif)

(a) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

(b) $A \cup (B \cap C) = (A \cup B) (A \cup C)$.

Bukti:

(a) Salah satu cara untuk menunjukkan bahwa dua himpunan adalah sama adalah dengan menetapkan bahwa masing-masing terdapat dalam himpunan lainnya. Misalkan t adalah elemen dari $A \cap (B \cup C)$. Maka t adalah elemen A dan t adalah elemen B atau elemen C. Dalam kedua kasus tersebut, t adalah elemen $A \cap B$ atau $A \cap C$. Dengan kata lain, $A \cap (B \cup C)$ adalah himpunan bagian dari $(A \cap B) \cup (A \cap C)$. Selanjutnya, misalkan t adalah elemen dari $(A \cap B) \cup (A \cap C)$. Ini menyiratkan bahwa t ada di $A \cap B$ atau di $A \cap C$. Jadi t pasti ada di A dan setidaknya ada di salah satu dari dua himpunan B atau C. Jadi t ada di A dan juga di B atau di C Dengan kata lain, t termasuk perpotongan dari A dan $B \cup C$. Jadi $(A \cap B) (A \cap C)$ adalah himpunan bagian dari $A \cap (B \cup C)$.

(b) Ini dibiarkan sebagai latihan.

TEOREMA 1.2 (Hukum De Morgan)

(a) $(A \cap B)^c = A^c \cup B^c$.

(b) $(A \cup B)^c = A^c \cap B^c$.

Bukti:

(a) Misalkan t adalah elemen dari $(A \cup B)^c$. Maka t bukan milik A atau B. Jadi t pasti ada di Ac dan Bc. Jadi $(A \cup B)^c$ adalah himpunan bagian dari perpotongan A^c dan B^c.

Sebaliknya, jika t berada pada perpotongan A^c dan B^c, maka t tidak berada di A maupun B. Hal ini berarti bahwa t tidak merupakan gabungan dari A dan B. Oleh karena itu, perpotongan A^c dan B^c terdapat dalam komplemen dari A ∪ B.

(b) Ini dibiarkan sebagai latihan.

1.2 FUNGSI DAN HUBUNGAN

Pada bagian ini disajikan tinjauan singkat tentang ide-ide dasar yang melibatkan fungsi dan relasi. Konsep fungsi sangat penting dalam matematika.

Domain dan Rentang Fungsi

Misalkan X dan Y adalah dua himpunan tak kosong. Fungsi f dari X ke Y, dilambangkan dengan f: X → Y, adalah aturan yang menetapkan setiap elemen di X elemen unik di Y. Himpunan X adalah domain dari fungsi dan himpunan Y adalah kodomainnya. Jika y adalah elemen unik dalam Y yang ditetapkan oleh fungsi f ke elemen x, kita katakan bahwa y adalah bayangan dari x dan x adalah bayangan dari y dan kita tulis y = f(x). Himpunan f(A) dari semua bayangan dari elemen-elemen subset A dari X disebut bayangan dari himpunan A. Himpunan f(X) disebut jangkauan fungsi. Jangkauan suatu fungsi adalah himpunan bagian dari kodomainnya. Jika y adalah elemen dalam jangkauan f, himpunan semua bayangan awal dari y dilambangkan dengan f^{-1} (y). Jika A adalah himpunan bagian dari jangkauan f(X), bayangan terbalik dari himpunan A adalah himpunan {x : x di X dan f(x) di A}, yang dilambangkan dengan f^{-1}(A) . Jika f adalah fungsi dari X ke Y, biasanya dikatakan bahwa f memetakan himpunan X ke Y.

Contoh 1.2.1

Misalkan R adalah himpunan semua bilangan real.

(a) Misalkan f: R → R adalah fungsi yang memberikan bilangan real x + 1 untuk setiap bilangan real x. Dengan kata lain, f(x) = x + 1. Di sini domain, kodomain, dan range dari f adalah R.

(b) Misalkan f: R → R adalah fungsi yang didefinisikan oleh f(x) = x^2. Jadi setiap bilangan real ditugaskan ke kuadratnya. Di sini domain dan kodomain dari f adalah R dan jangkauannya adalah himpunan semua bilangan nonnegatif.

Contoh 1.2.2

Misalkan A = {a, b, c} dan B = {1, 2, 3, 4}. Maka aturan f yang didefinisikan oleh f(a) = 1, f(b) = 1, f(c) = 4, dan f(d) = 2 adalah fungsi f dari A ke B. Jangkauan f adalah {1 , 2, 4}, yang merupakan himpunan bagian yang tepat dari kodomain B-nya.

Surjeksi, Suntikan, dan Bijeksi

Suatu fungsi f: X → Y disebut surjeksi jika f(X) = Y dan kita katakan bahwa f adalah fungsi dari X ke Y. Fungsi f: X → Y disebut injeksi (atau pemetaan satu-satu) jika dua elemen berbeda di X memiliki dua bayangan berbeda di Y. Fungsi f: X → Y adalah bijeksi jika keduanya merupakan surjeksi dan injeksi. Bijeksi dari himpunan X ke dirinya sendiri yang memetakan setiap elemen dalam himpunan ke dalam dirinya sendiri disebut pemetaan identitas ix pada X. Dua himpunan ekuivalen jika ada bijeksi dari satu ke yang lain. Jelaslah bahwa dua himpunan hingga adalah ekuivalen jika dan hanya jika keduanya memiliki kardinalitas yang sama.

Contoh 1.2.3

(a) Misalkan X = {a, b, c}, Y = {p, q}, dan f: X → Y, di mana f(a) = p, f(b) = q, dan f(c) = p. Maka f adalah surjeksi dan f memetakan X ke Y. Di sini f bukan injeksi.
(b) Jika X = {a, b, c}, Y = {p, q, r, s} dan jika g(a) = p, g(b) = q, g(c) = r, maka g adalah injeksi tetapi bukan surjeksi. Rentang g(X) = {p, q, r} adalah himpunan bagian yang tepat dari kodomain Y.
(c) Jika X = {a, b, c}, Y = {p, q, r} dan jika h(a) = p, h(b) = q, dan h(c) = r, maka h adalah sebuah bijeksi.
(d) Jika R adalah himpunan bilangan real dan f: R → R fungsi yang didefinisikan oleh f(x) = x^2, maka f bukan surjeksi, karena tidak ada bilangan negatif yang memiliki pracitra, atau injeksi, karena citra dari x dan bayangan –x keduanya sama.

Invers suatu Fungsi

Misalkan f: X → Y adalah bijeksi. Fungsi invers dari f adalah bijeksi f–1: Y → X didefinisikan sebagai berikut: Untuk setiap y dalam Y, kita temukan elemen unik x dalam X sedemikian sehingga f(x) = y. Kemudian kita definisikan x = $f^{-1}(y)$. Suatu fungsi f: X → Y dikatakan invertible apabila inversnya ada.

Contoh 1.2.4

Jika X = {1, 2}, Y = {p, q}, f(1) = p, dan f(2) = q, maka f adalah bijeksi dari X ke Y dan inversnya f-1 adalah bijeksi dari Y ke X yang memetakan p menjadi 1 dan q menjadi 2.

Suatu fungsi f yang domainnya X dan kodomain Y adalah himpunan bagian dari himpunan R bilangan real meningkat secara ketat jika f(x) < f(y) setiap kali x < y dan sangat menurun jika f(x) > f(y) setiap kali x < y. Ini mengikuti dari definisi bahwa baik fungsi yang meningkat secara ketat dan fungsi yang sangat menurun adalah suntikan.

Komposisi Fungsi

Misal g : X → Y dan f: Y → Z. Komposisi f dan g, didefinisikan oleh f g, adalah fungsi dari X ke Z yang didefinisikan oleh (f g)(x) = f(g(x)). Dengan kata lain, fungsi f g menetapkan ke elemen x di X elemen unik yang diberikan oleh f ke g(x).

Contoh 1.2.5

(a) Misalkan X = {a, b, c}, Y = {p, q, r, s}, dan Z = {1, 2, 3}. Misalkan g(a) = p, g(b) = q, dan g(c) = r, sehingga g(X) = {p, q, r}. Kemudian jika f: g(X) → Z didefinisikan oleh f(p) = 1, f(q) = 2, dan f(r) = 3, kita dapatkan:

$$(f \circ g)(a) = f(g(a)) = f(p) = 1$$
$$(f \circ g)(b) = 2$$
$$(f \circ g)(c) = 3$$

(b) Misalkan f dan g adalah fungsi dari himpunan bilangan bulat ke himpunan bilangan bulat. Jika f(x) = 4x + 3 dan g(x) = 2x + 5, kemudian:

$$(f \circ g)(x) = f(g(x)) = f(2x + 5) = 4(2x + 5) + 3 = 8x + 23$$
$$(g \circ f)(x) = g(f(x)) = g(4x + 3) = 2(4x + 3) + 5 = 8x + 11$$

Jika f adalah bijeksi dari X ke Y, inversnya adalah bijeksi dari Y ke X. Jika y = f(x), maka $f^{-1}(y)$ = x. Jadi $f^{-1}(f(x) = f^{-1}(y) = x$ dan $f(f^{-1}(y)) = f(x) = y$. Dengan kata lain, komposisi bijeksi dari X ke Y dan kebalikannya adalah pemetaan identitas dari Y ke dirinya sendiri.

Urutan, String, dan Bahasa

Barisan adalah suatu fungsi yang domainnya merupakan himpunan bilangan bulat berurutan. Jika domain X adalah himpunan berhingga dari n bilangan bulat, kita dapat mengambil X = {1, 2, 3, . . . , n} atau {0, 1, 2, . . . , n – 1}. Jika tidak, kita dapat mengambil X sebagai himpunan bilangan asli atau sebagai himpunan bilangan bulat nonnegatif. Jika f: X → Y adalah barisan, bayangan f(i) dari bilangan bulat i kadang-kadang ditulis sebagai fi dan disebut suku ke-i dari barisan tersebut.

Perhatikan bahwa dalam mewakili urutan s, urutan munculnya gambar di bawah s adalah penting. Hal ini tidak terjadi dalam kasus fungsi. Misalnya, jika f adalah fungsi dari X = {1, 2, 3} ke Y = {p, q}, di mana f(1) = f(2) = p dan f(3) = q, himpunan dari gambar dari tiga elemen X di bawah f dapat direpresentasikan sebagai p, p, q dalam urutan apa pun. Tetapi barisan f direpresentasikan sebagai (f(1)f(2)f(3)) atau sebagai (ppq). Suatu barisan yang domainnya berhingga yang terdiri dari n bilangan bulat berurutan dan yang kodomainnya adalah Y mendefinisikan string dengan panjang n dalam Y atau kata dengan panjang n dalam Y. Faktanya, setiap barisan seperti itu adalah n-tupel.

Contoh 1.2.6

(a) Misalkan X = {1, 2, 3, . . .} dan R himpunan bilangan real. Perhatikan barisan f: X → R yang didefinisikan oleh f(n) = 1/n. Maka suku ke-n dari barisan yang dilambangkan dengan fn adalah bayangan f(n) dari elemen n pada X. Barisan ini juga dilambangkan dengan {1/n : n = 1, 2, 3, . . .}.

(b) Misalkan X = {1, 2, 3, 4, 5} dan Y = {a, b, c, d} dan perhatikan barisan f: X → Y didefinisikan oleh f(1) = a, f(2) = b, f(3) = a, f(4) = c, dan f(5) = b. Maka barisan ini adalah string abacb dengan panjang 5 di Y yang juga merupakan 5-tuple (abacb).

Pemetaan f dari A × A ke A disebut operator biner pada A. Misalnya, jika R adalah himpunan bilangan real, pemetaan f: R × R → R didefinisikan oleh f(a, b) = a + b (yang sebenarnya adalah operator penjumlahan) adalah contoh dari operator biner pada R.

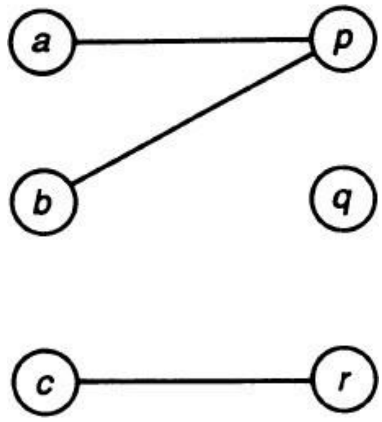
Jika S adalah sembarang himpunan tak kosong, kita nyatakan dengan Sn himpunan semua string dengan panjang n dalam S dan S* himpunan semua string (termasuk string nol tanpa elemen). Setiap subset dari S* disebut bahasa di atas alfabet S. Persatuan dan perpotongan dari dua bahasa di atas alfabet juga bahasa di atas alfabet yang sama. Jika u = $(u_1u_2u_3 \cdots u_m)$ dan v = $(v_1v_2 \cdots v_n)$ adalah dua dawai dengan panjang m dan n, berturut-turut di S* maka rangkaian u dan v adalah dawai uv di S* dengan panjang m + n didefinisikan sebagai uv = $(u_1u_2u_3 \cdots u_mv_1v2 \cdots v_n)$. Pemetaan c: S* × S* → S* didefinisikan oleh c(u, v) = uv dimana uv adalah gabungan dari u dan v adalah operator biner pada S*.

Hubungan

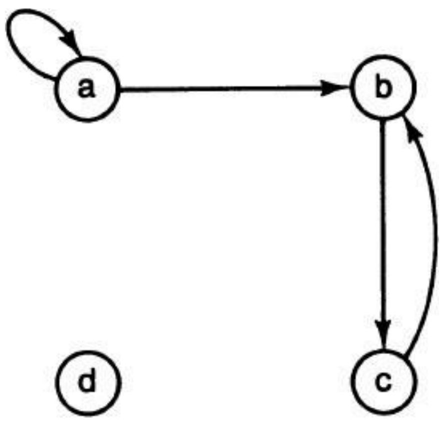
Kami menyimpulkan bagian ini dengan komentar singkat tentang konsep "hubungan", yang lebih umum daripada fungsi. Jika A dan B adalah dua himpunan, setiap himpunan bagian dari A × B disebut relasi dari A ke B. Misalnya, jika A = {a, b, c} dan B = {1, 2, 3, 4}, maka R = {(a, 2), (a, 3), (b, 4), (c, 3)} adalah relasi dari A ke B. Menurut definisi, pada setiap pasangan terurut dalam relasi dari A ke B, elemen pertama adalah elemen dalam A dan elemen kedua adalah elemen dalam B. Oleh karena itu, suatu fungsi dari A ke B mendefinisikan jenis relasi khusus R dari A ke B sedemikian sehingga setiap kali (a, b) dan (a, b′) berada dalam relasi R, maka b = b′. Dengan kata lain, f: A → B mendefinisikan produk kartesius {(x, f(x)) : x ada di A}, yang merupakan subset dari A × B.

Suatu relasi R dari himpunan berhingga A dengan m elemen ke himpunan berhingga dengan n elemen dapat direpresentasikan secara piktorial oleh graf bipartit G dengan m simpul di sisi kiri (sesuai dengan m elemen A) dan n simpul di sisi kanan (sesuai dengan n elemen B) seperti pada Gambar 1.2. Jika (a, p) adalah elemen dalam relasi R, sebuah panah ditarik dari titik a di sisi kiri ke titik p di sisi kanan. Sebagai contoh, grafik pada Gambar 1.2 merepresentasikan relasi R = {(a, p), (b, p), (c, r)} dari himpunan A = {a, b, c} ke himpunan B = {p, q, r}.

Suatu relasi dari himpunan A ke dirinya sendiri disebut relasi pada A. Cara yang informatif dan berguna untuk merepresentasikan relasi R pada himpunan berhingga A dengan n elemen adalah dengan menggambar graf berarah dengan n simpul yang mewakili n elemen dari himpunan tersebut dan menggambar panah dari simpul u ke simpul v jika dan hanya jika pasangan terurut (u, v) ada dalam relasi. Jika (u, u) ada dalam relasi, sebuah loop dari u ke u dibuat. Misalnya, jika R = {(a, a), (a, b), (b, c), (c, b)} adalah relasi pada himpunan A = {a, b, c}, relasi ini R dapat diwakili oleh grafik berarah yang ditunjukkan pada Gambar 1.3.



Gambar 1.2 Grafik relasi R, himpunan a dan himpunan b



Gambar 1.3 Relasi R dapat diwakili oleh grafik berarah

Suatu relasi R pada A adalah refleksif jika (a, a) adalah elemen R untuk setiap a di A, simetris jika (a, b) berada di R setiap kali (b, a) ada di R dan transitif jika (a, c) ada di R jika (a,

b) dan (b, c) ada di R. Suatu relasi R pada suatu himpunan adalah antisimetris jika setiap a dan b berbeda, maka (a, b) ada dalam relasi R hanya jika (b, a) tidak ada dalam relasi R. Akhirnya, relasi R dikatakan memiliki sifat pembanding jika (a, b) atau (b, a) ada di R untuk semua a dan b di A .

Misalkan R adalah relasi pada himpunan berhingga A dan misalkan G adalah graf berarah yang merepresentasikan relasi tersebut. Maka R refleksif jika dan hanya jika ada loop di setiap simpul G dan R simetris jika dan hanya jika setiap kali ada panah dari a ke b, ada panah dari b ke a. Selanjutnya, R transitif jika dan hanya jika setiap kali ada panah dari a ke b dan panah dari b ke c ada panah dari a ke c.

Contoh 1.2.7

Misalkan A = {a, b, c} dan misalkan R suatu relasi pada A.

(a) R = {(a, b), (b, a), (a, a), (b, b), (b, c), (c, c)} bersifat refleksif karena (u, u) ada di R untuk semua u di A. Di sini (a, a), (b, b), dan (c, c) ada di R. Ketiga elemen ini akan mewakili loop pada tiga simpul dari digraf yang sesuai.

(b) R = {(a, b), (b, a), (c, c)} simetris karena setiap kali (u, v) ada di R untuk setiap u dan setiap v di A, maka (v, u) juga ada di R. Di sini baik (a, b) dan (b, a) serta (c, c) ada di R. Dalam digraf yang mewakili hubungan ini akan ada panah dari a dan b dan dari b ke sebuah. Akan ada loop di simpul c.

(c) R = {(a, b), (b, c), (a, c), (b, b)} transitif.

(d) R = {(a, c), (b, b), (a, b), (a, a)} antisimetris.

(e) Jika R = {(a, c), (b, b), (c, c), (a, b), (c, b)}, maka R memiliki sifat pembanding.

Hubungan Kesetaraan

Suatu relasi S pada suatu himpunan disebut relasi ekivalensi pada A jika S bersifat refleksif, simetris, dan transitif. Misalnya, jika S = {(a, b) : a, b real, a = b}, maka S jelas merupakan relasi ekivalen pada himpunan bilangan real.

Contoh 1.2.8

(a) Misalkan A = {a, b, c, d, e} dan C(A) adalah partisi dari A yang didefinisikan oleh kelas {{a, b}, {c, d, e}}. Misalkan R adalah himpunan pasangan terurut (x, y) pada A × A sehingga setiap kali x berada pada salah satu himpunan pada partisi, maka y juga berada pada himpunan yang sama. Jadi dalam hal ini R = {(a, a), (b, b), (c, c), (d, d), (e, e), (a, b), (b, a), (c, d), (d, c), (c, e), (e, c), (d, d), (d, e)}. Sangat mudah diverifikasi bahwa R adalah relasi ekivalensi. Setiap partisi dari suatu himpunan mendefinisikan relasi ekivalensi unik di dalamnya.

(b) Sebaliknya, dapat dengan mudah ditentukan bahwa setiap relasi ekivalensi pada suatu himpunan mendefinisikan partisi pada himpunan tersebut. Jika pasangan terurut (a, b) termasuk dalam relasi ekivalensi pada himpunan A, kita ambil a dan b merupakan subset yang sama dari A. Kelas dari subset yang terbentuk merupakan partisi dari X. Misalnya relasi ekivalensi R = {(p, p), (q, q), (p, q), (q, p), (r, r)} mendefinisikan partisi {{p, q}, {r}} dari himpunan {p , q, r}.

Set Kesetaraan dan Kelas Kesetaraan

Misalkan R suatu relasi ekivalen pada himpunan A dan misalkan x adalah sembarang elemen dari A. Himpunan ekivalen [x] dari elemen x adalah himpunan {y : (y, x) R}. Amati bahwa jika [u] dan [v] adalah dua himpunan ekivalen yang berbeda, perpotongannya kosong.

Karena jika x ada di keduanya [u] dan di [v], maka karena transitivitas (u,v) ada pada relasi R yang berimplikasi [u] = [v]. Kelas himpunan ekuivalen berbeda dari elemen-elemen dalam X disebut kelas ekivalen dari relasi tersebut. Kelas ekivalen dari suatu himpunan adalah partisi dari suatu himpunan, dan sebaliknya. Jadi tidak ada perbedaan nyata antara partisi dari suatu himpunan dan kelas ekivalensi dalam himpunan tersebut. Dalam prakteknya, hampir selalu terjadi bahwa kita menggunakan relasi ekivalensi untuk mendapatkan partisi karena biasanya mudah untuk mendefinisikan sebuah relasi ekivalensi pada suatu himpunan.

Perintah Parsial dan Perintah Linear

Suatu relasi R pada A adalah orde parsial jika relasi tersebut refleksif, antisimetris, dan transitif. Orde parsial R yang memiliki sifat perbandingan disebut orde total (atau linier). Himpunan tak kosong A bersama-sama dengan relasi orde parsial P yang didefinisikan padanya disebut himpunan terurut sebagian (set PO) dan dilambangkan dengan (A, P). Himpunan terurut sebagian (A, P) disebut himpunan terurut total (linier) atau rantai jika P memiliki sifat pembanding.

Contoh 1.2.9

(a) Misalkan A adalah himpunan tak kosong dan P(A) himpunan dayanya. Misalkan R adalah relasi pada P(A) × P(A) yang didefinisikan oleh properti “set-inklusi”; yaitu, jika E dan F adalah himpunan bagian dari A, maka (E, F) berada pada relasi R jika E adalah himpunan bagian dari F. Maka R adalah orde parsial pada P(A) dan (P(A), R) adalah himpunan terurut sebagian. Tapi itu bukan himpunan terurut linier untuk subset arbitrer dari A, tidak perlu mengandung subset arbitrer A lainnya.

(b) Jika x dan y adalah dua bilangan real, kita katakan bahwa (x, y) adalah elemen dalam relasi S pada himpunan R bilangan real setiap kali x lebih kecil dari atau sama dengan y. Maka relasi S adalah orde linier pada R.

Contoh 1.2.10

Misalkan X = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} dan S adalah relasi pada X yang didefinisikan sebagai S = {(m, n) : m membagi n}. Maka S adalah orde parsial pada S. Himpunan A = {2, 4, 8} adalah rantai di X, sedangkan himpunan B = {2, 5, 10} bukan rantai karena elemen 2 dan 5 tidak sebanding.

Diagram Hasse dari Himpunan Terurut Sebagian

Perhatikan graf berarah G yang merepresentasikan orde parsial R pada himpunan berhingga A. Karena R refleksif, ada loop di setiap simpul dari graf tersebut. Karena R transitif, ada busur dari simpul u ke simpul v setiap kali ada busur dari u ke w dan busur dari w ke v. Jadi kita dapat memiliki representasi bergambar yang disederhanakan dari orde parsial jika kita mengabaikan loop dan hapus semua panah yang ada karena transitivitas. Selanjutnya, jika representasi grafis sangat berorientasi sehingga semua panah menunjuk ke satu arah (atas, bawah, kiri ke kanan, atau kanan ke kiri), kita dapat mengabaikan arah panah juga. Diagram yang dihasilkan disebut diagram Hasse dari himpunan terurut sebagian.

Contoh 1.2.11

Misal X = {1, 2, 3, 4, 5} dan S = {(1, 1), (1, 2), (1, 3), (1,4), (1, 5), (2 , 2), (2, 3), (2, 5), (3, 3), (3, 5), (4, 4), (4, 5), (5, 5)}. Dapat dengan mudah diverifikasi bahwa S adalah orde parsial pada X. Diagram Hasse yang mewakili S ditunjukkan pada Gambar 1.5.

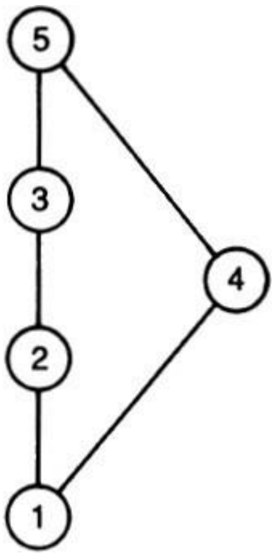
Elemen Maksimal dan Minimal

Suatu elemen u pada himpunan A yang terurut sebagian dengan orde parsial R disebut elemen maksimal pada himpunan tersebut jika setiap kali (u, x) berada di R, maka x = u. Demikian pula, elemen v di A adalah elemen minimal jika setiap kali (x, v) ada di R, maka x = v.

Contoh 1.2.12

Misalkan X = {2, 3, 4, 5, 8, 12, 24, 25} dan misalkan R adalah orde parsial pada X yang didefinisikan oleh R = {(m, n) : m membagi n}. Maka 2 adalah elemen minimal dari R karena tidak ada elemen di X yang membagi 2. Demikian pula, 3 dan 5 juga merupakan elemen minimal dari R. Demikian pula, 24 adalah elemen maksimal karena tidak ada bilangan di X yang habis dibagi 24. Maksimal lainnya elemen dalam X adalah 25.

Elemen minimal dan maksimal dari orde parsial dapat dengan mudah terlihat menggunakan diagram Hasse-nya, di mana elemen minimal akan berada di bawah dan elemen maksimal akan berada di atas jika semua panah ditarik ke atas. Lihat Gambar 1.4, yang mewakili diagram Hasse dari Contoh 1.2.12, di mana simpul yang mewakili 24 dan 25 berada di atas dan simpul yang mewakili 2, 3, dan 5 berada di bawah.

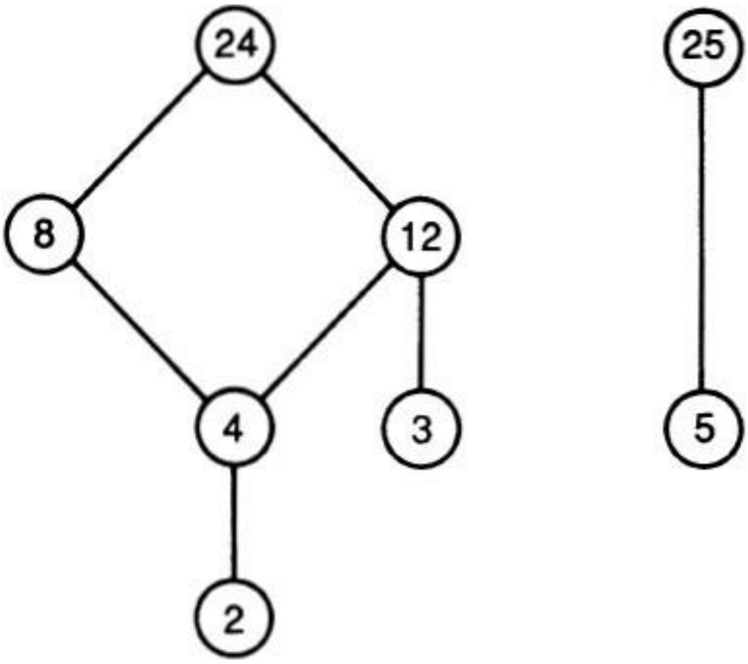


Gambar 1.4 Grafik yang mewakili diagram hasse

Contoh 1.2.13

P(A) adalah himpunan terurut sebagian dari semua himpunan bagian A dengan orde parsial ditentukan oleh inklusi himpunan, dan dalam himpunan PO ini, A adalah satu-satunya elemen maksimal dan himpunan kosong adalah satu-satunya elemen minimal.

Himpunan terurut sebagian mungkin memiliki lebih dari satu maksimal (atau minimal), seperti yang kita lihat pada Contoh 1.2.12. Ada himpunan terurut sebagian tanpa elemen maksimal atau minimal. Perhatikan relasi S = {(x, y) : x, y bilangan bulat, x y}. Maka S tidak diragukan lagi merupakan orde parsial pada himpunan Z bilangan bulat, tetapi himpunan PO ini tidak memiliki elemen maksimal atau minimal.



Gambar 1.5 Diagram hasse

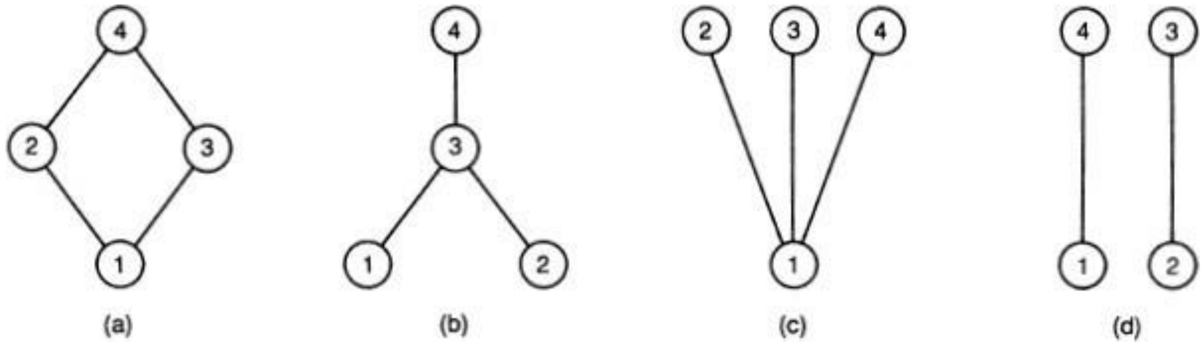
Elemen Maksimum (Terbesar) dan Minimum (Terkecil)

Suatu elemen M dalam himpunan A yang terurut sebagian dengan orde parsial S disebut maksimum (atau elemen terbesar) di A jika (x, M) ∈ S untuk setiap x dalam himpunan A. Demikian pula, elemen m adalah minimum (atau elemen terkecil) jika (m, x) ∈ S untuk setiap x himpunan A.

[Seseorang harus sangat berhati-hati dalam membedakan (1) antara elemen maksimal dan elemen maksimum dan (2) antara elemen minimal dan elemen minimal. Jika suatu elemen adalah maksimum atau minimum, semua elemen dalam himpunan harus sebanding dengannya. Tentu saja, jika ada elemen maksimum, tidak diragukan lagi itu adalah elemen maksimal. Demikian pula, jika ada elemen minimum, itu adalah elemen minimal. Implikasi sebaliknya belum tentu benar, seperti dapat dilihat dari diagram Hasse pada Contoh 1.2.14. Dalam pemerintahan multipartai, setiap pemimpin partai dapat dianggap sebagai elemen maksimal, sedangkan dalam sistem satu partai, pemimpin partai yang unik adalah maksimal dan maksimal.]

Contoh 1.2.14

Misalkan A = {1, 2, 3, 4} dan perhatikan empat orde parsial pada A dengan diagram Hasse seperti pada Gambar 1.6. Pada bagian (a), 4 adalah elemen terbesar dan 1 adalah elemen terkecil. Pada (b), 4 adalah elemen terbesar dan elemen terkecil adalah 1 dan 2. Tidak ada elemen terkecil pada (b). Dalam (c), 1 adalah elemen terkecil. Tidak ada elemen terbesar di sini; tetapi 2, 3, dan 4 adalah elemen maksimal. Tidak ada elemen terbesar atau terkecil dalam (d). Elemen 1 dan 2 minimal dan elemen 3 dan 4 maksimal.



Gambar 1.6 Elemen maksimum dan minimum

Contoh 1.2.15

Dalam setiap himpunan bilangan bulat positif berikut, pasangan terurut (m, n) berada dalam relasi S jika m membagi n.

(a) A = {2, 4, 6, 8}. Di sini 2 adalah elemen terkecil. Tidak ada elemen terbesar. Elemen maksimalnya adalah 6 dan 8.

(b) A = {2, 3, 4, 12}. Unsur terbesar adalah 12. Tidak ada unsur terkecil. Elemen minimalnya adalah 2 dan 3.

(c) A = {2, 4, 8, 16}. Yang terbesar adalah 16 dan yang terkecil adalah 2.

(d) A = {2, 3, 4, 6}. Di sini 2 dan 3 minimal; 4 dan 6 maksimal.

Set yang Dipesan dengan Baik

Himpunan terurut sebagian A di mana setiap himpunan bagian B memiliki elemen terkecil m di B disebut himpunan terurut baik. Sebagai contoh, jika N adalah himpunan semua bilangan bulat positif, dan jika kita mengatakan bahwa (a, b) ada di S setiap kali a lebih kecil dari atau sama dengan b, maka S adalah orde parsial pada N. Misalkan B sembarang himpunan bagian dari N. Jelas, bilangan bulat terkecil di B adalah elemen terkecil dari B.

Jadi setiap himpunan bagian dari N memiliki elemen terkecil di dalamnya. Jadi N adalah himpunan yang terurut dengan baik. Himpunan A bilangan real dalam suatu interval bukanlah himpunan yang tertata baik di bawah relasi S, di mana (x, y) dalam S berarti x lebih kecil atau sama dengan y. Subset dari himpunan yang tertata dengan baik adalah yang tertata dengan baik. Himpunan A yang terurut dengan baik adalah terurut linier karena setiap himpunan dari dua elemen di A memiliki elemen pertama dan oleh karena itu relasinya memiliki sifat perbandingan.

Lemma Zorn

Misalkan B adalah himpunan bagian dari himpunan A yang terurut sebagian dengan relasi orde parsial S. Suatu elemen u di A disebut batas atas B jika (x, u) ada di S untuk semua x di B. Perhatikan bahwa u tidak perlu berada di B. Jika ada batas atas untuk B, kita katakan bahwa B memiliki batas atas. Subset sembarang dari set PO tidak perlu memiliki batas atas.

Salah satu alat matematika yang paling penting dan sangat kuat adalah lemma Zorn, yang menyatakan bahwa jika A adalah himpunan terurut sebagian di mana setiap rantai memiliki batas atas, maka A memiliki elemen maksimal. Lemma ini tidak dapat "dibuktikan" dalam pengertian istilah yang biasa. Namun, dapat ditunjukkan bahwa itu secara logis setara dengan aksioma pilihan yang terkenal, yang terletak di dasar teori himpunan. Dengan demikian lemma Zorn diasumsikan sebagai aksioma logika dan teori himpunan. Banyak teorema keberadaan yang penting dibuktikan dengan menggunakan lemma Zorn. Aksioma pilihan juga secara logis setara dengan teorema keteraturan baik dari Zermelo: Setiap himpunan dapat diurutkan dengan baik. Untuk bukti lemma Zorn, menggunakan aksioma pilihan, lihat buku Teori Himpunan Naif oleh P. R. Halmos.

1.3 BUKTI INDUKTIF DAN DEFINISI REKURSIF

Salah satu teknik pembuktian yang paling berguna, elegan, dan sederhana dalam matematika pada umumnya dan matematika diskrit pada khususnya adalah teknik yang dikenal sebagai induksi matematika, yang pada dasarnya adalah "prosedur pembuktian algoritmik." Asal usul teknik ini dapat ditelusuri hingga zaman periode Yunani klasik. Tetapi istilah induksi diciptakan oleh De Morgan hanya pada abad kesembilan belas.

Prinsip Induksi Matematika (Bentuk Lemah)

Prinsip ini dinyatakan sebagai berikut: Misalkan P(n) adalah pernyataan tentang bilangan asli n dan q adalah bilangan asli tetap. Maka pembuktian induksi bahwa P(n) benar untuk semua $n \geq q$ memerlukan dua langkah:

1. Langkah dasar: Verifikasi bahwa P(q) benar.
2. Langkah induksi: Pastikan bahwa jika k adalah bilangan bulat positif yang lebih besar dari atau sama dengan q, maka P(k + 1) benar jika P(k) benar.

Di sini P(n) disebut hipotesis induktif. Ketika kita menyelesaikan kedua langkah pembuktian dengan induksi matematika, jelas bahwa kita telah membuktikan bahwa pernyataan P(n) benar untuk semua bilangan bulat positif n q. [Sebuah bukti formal mengikuti baris berikut: Misalkan P(n) tidak benar untuk semua $n \geq q$. Maka paling sedikit ada satu bilangan bulat positif k q sedemikian sehingga P(k) tidak benar. Jadi himpunan D dari bilangan bulat positif r yang P(r) tidak benar adalah tidak kosong, dan oleh karena itu himpunan ini memiliki elemen terkecil yang unik karena setiap himpunan bilangan bulat positif terurut dengan baik. Biarkan t menjadi elemen pertama dari himpunan ini. Tentu saja, t > q. Karena t adalah bilangan bulat, t – 1 juga merupakan bilangan bulat yang tidak ada di D. Jadi P(t – 1) benar, yang menyiratkan dengan langkah induksi bahwa P(t) benar. Ini adalah kontradiksi.]

Versi induksi yang diberikan di atas disebut bentuk lemah karena langkah induksi mengasumsikan bahwa P(n) benar dalam tepat satu kasus. Bentuk induksi yang kuat dibahas kemudian di bagian ini.

Contoh 1.3.1

Buktikan bahwa $1 + 2 + 3 + \cdots + n = n(n + 1)/2$ untuk semua bilangan asli.

Bukti (Dengan Induksi Matematika):

Misalkan P(n) adalah pernyataan bahwa $(1 + 2 + 3 + \cdots + n)$ sama dengan $n(n + 1)/2$. Tujuannya adalah untuk membuktikan bahwa P(n) benar untuk semua n.

Langkah dasar:

Kita harus memverifikasi bahwa P(1) benar. Di sini P(1) adalah pernyataan bahwa 1 sama dengan $1(1 + 1)/2$. Ini benar. Jadi P(1) benar.

Langkah induksi:

Kita harus memverifikasi bahwa jika P(k) benar, maka P(k + 1) benar. Sekarang P(k) adalah pernyataan bahwa $1 + 2 + \cdots + k$ sama dengan $k(k + 1)/2$ dan P(k + 1) adalah pernyataan bahwa $1 + 2 + \cdots + (k + 1)$ sama dengan $(k + 1)(k + 2)/2$. Jika P(k) benar, $1 + 2 + \cdots + k = k(k + 1)/2$. Jadi

$1 + 2 + \cdots + k + (k + 1) = k(k + 1)/2 + (k + 1)$, yang berarti bahwa $1 + 2 + \cdots + (k + 1) = (k + 1) \bullet (k + 2)/2$. Jadi P(k + 1) benar jika P(k) benar.

Seseorang harus sangat berhati-hati dalam menggunakan metode induksi untuk membuktikan teorema. Arti dari langkah induksi sangat tepat: Jika P(q) benar, P(q + 1) benar. Jika P(q + 1) benar, maka P(q + 2) benar, dan seterusnya. Dengan kata lain, induksi harus berlaku untuk setiap k lebih besar dari atau sama dengan q. Sebuah "bukti" yang salah membenarkan pernyataan bahwa semua mawar memiliki warna yang sama adalah sebagai berikut. P(n) adalah proposisi bahwa semua mawar dalam koleksi n mawar memiliki warna yang sama. Tujuan kita adalah untuk menunjukkan bahwa P(n) benar untuk semua n. Jelas, P(1) benar. Misalkan P(k) benar untuk beberapa bilangan bulat positif k. Jadi semua mawar dalam koleksi k roses memiliki warna yang sama. Sekarang pertimbangkan koleksi sembarang C dari (k + 1) mawar. Beri label mawar ini sebagai ri (i = 1, 2, 3, . . , k + 1). Biarkan A menjadi himpunan $\{r_i : i = 1, 2, . . . , k\}$ dan B adalah himpunan $\{r_i : i = 2, 3, . . . , k, k + 1\}$. Baik A dan B memiliki tepat k mawar. Jadi mawar di A semuanya berwarna sama. Demikian pula, mawar di B juga memiliki warna yang sama. Mawar berlabel rk ada di A dan B. Jadi semua mawar k + 1 yang dipertimbangkan memiliki warna yang sama. Jadi jika P(k) benar, maka P(k + 1) benar. Oleh karena itu, dapatkah kita menyimpulkan bahwa P(n) benar untuk semua n? Jawabannya "tidak" karena jika himpunan C memiliki dua anggota, maka himpunan A dan B saling lepas. Jadi jika P(1) benar, P(2) tidak perlu benar.

Contoh 1.3.2

Gunakan induksi untuk membuktikan bahwa jumlah n bilangan bulat positif ganjil pertama adalah n^2.

Bukti:

Misalkan P(n) adalah pernyataan bahwa jumlah n bilangan bulat positif ganjil pertama adalah n^2. Tujuannya adalah untuk membuktikan bahwa P(n) benar untuk setiap n.

Langkah dasar:

P(1) benar karena 1 = 12.

Langkah induksi:

Kita harus memverifikasi bahwa P(k + 1) benar setiap kali P(k) benar untuk sembarang bilangan bulat positif k.

Karena P(k) benar, $1 + 3 + 5 + \cdots + (2k – 1) = k2$. Akibatnya:

$$1 + 3 + 5 + \cdots + (2k - 1) + (2k + 1) = k^2 + (2k + 1) = (k + 1)^2$$

Jadi P(k+1) benar. Jadi P(n) benar untuk setiap n.

Contoh 1.3.3

Gunakan induksi untuk membuktikan bahwa $n < 2^n$ untuk sembarang bilangan bulat positif n.

Bukti:

Misalkan P(n) adalah pernyataan bahwa $n < 2^n$ untuk bilangan bulat positif n. Kita harus membuktikan bahwa P(n) benar untuk sembarang bilangan bulat positif.

Langkah dasar:

P(1) benar karena 1 < 2.

Langkah induksi:

Misalkan P(k) benar untuk bilangan bulat positif sembarang, yang menyiratkan bahwa $k < 2^k$. Maka $k+1 < 2^k + 1 < 2^k + 2^k$. Oleh karena itu $k + 1 < 2^k + 1$, yang menyiratkan bahwa P(k + 1) benar. Jadi P(n) benar untuk semua bilangan bulat positif n.

Prinsip Induksi Matematika (Bentuk Kuat)

Misalkan P(n) adalah pernyataan tentang bilangan asli n dan q adalah bilangan asli tetap. Maka bukti induksi bahwa P(n) benar untuk semua $n \geq q$ membutuhkan dua langkah:

1. Langkah dasar: Verifikasi bahwa P(q) benar.
2. Langkah induksi: Pastikan jika $k \geq q$ dan jika P(q), P(q + 1), P(q + 2), . . . , P(k) benar, maka P(k + 1) benar.

(Versi prinsip induksi ini "kuat" dalam arti bahwa langkah induksi di sini memiliki lebih banyak informasi daripada langkah induksi dalam versi "lemah". Seperti pada kasus sebelumnya, dapat dengan mudah ditunjukkan bahwa versi kuat juga merupakan konsekuensi dari fakta bahwa setiap himpunan bilangan asli terurut dengan baik. Jadi untuk membuktikan teorema menggunakan induksi matematika, seseorang dapat menggunakan salah satu versi induksi matematika. Dalam beberapa kasus lebih mudah menggunakan bentuk kuat, seperti terlihat pada contoh berikut. Bentuk kuat induksi juga dikenal sebagai induksi lengkap.)

Contoh 1.3.4

Buktikan bahwa bilangan asli apa pun yang lebih besar dari 1 dapat difaktorkan sebagai hasil kali bilangan prima.

Bukti (Dengan Induksi Matematika Lengkap):

Misalkan P(n) adalah pernyataan bahwa jika n adalah bilangan asli yang lebih besar dari 1, maka n dapat difaktorkan sebagai hasil kali bilangan prima. Tujuannya adalah untuk membuktikan bahwa P(n) benar untuk semua n.

Langkah dasarnya:

P(2) adalah pernyataan bahwa 2 dapat difaktorkan sebagai hasil kali bilangan prima. Jelas, P(2) benar.

Langkah induksi:

Misalkan P(2), P(3), . . . , P(k) benar. Kita harus memverifikasi bahwa P(k + 1) benar. Sekarang P(k + 1) pasti benar ketika k + 1 adalah bilangan prima. Jika k + 1 bukan bilangan prima, kita selalu dapat menemukan dua bilangan bulat positif m dan n sedemikian rupa sehingga k + 1 = mn, di mana m dan n lebih kecil dari k. Dengan langkah induksi, baik m dan n dapat dinyatakan sebagai hasil kali bilangan prima. Jadi k + 1 juga dapat difaktorkan sebagai hasil kali bilangan prima. Jadi P(k+1) benar.

Definisi Rekursif dari Himpunan

Ide dasar yang mendasari prinsip induksi adalah sebagai berikut. Setelah kami menggambarkan tahap awal dalam beberapa proses dan jika kami dapat menggambarkan tahap selanjutnya dalam hal tahap sebelumnya, kami berada dalam posisi untuk menggambarkan seluruh proses secara lengkap di semua tahap. Konsep paralel dalam ilmu komputer adalah rekursi, di mana kita cenderung memikirkan proses dalam arah yang berlawanan. Secara informal, ini adalah proses pemecahan masalah besar dengan menguraikannya menjadi satu atau lebih submasalah sehingga setiap submasalah tersebut identik dalam struktur dengan masalah aslinya tetapi lebih atau kurang lebih sederhana untuk dipecahkan. Jadi dalam kedua situasi, seseorang harus (1) memutuskan satu set kasus sederhana yang pembuktian atau perhitungannya mudah ditangani, dan (2) mendapatkan aturan yang sesuai yang dapat diterapkan berulang kali sampai akhir. Konsep yang mendasari induksi dan rekursi ini dapat digunakan untuk membenarkan definisi beberapa kumpulan objek secara bertahap. Deskripsi seperti itu disebut tepat definisi induktif atau rekursif.

Definisi rekursif dari suatu himpunan terdiri dari tiga bagian:

1. Bagian Basis: Bagian ini memberitahu kita bahwa elemen-elemen tertentu termasuk dalam himpunan yang akan kita definisikan.
2. Bagian induktif (rekursif): Bagian ini memberitahu kita untuk menggunakan elemen-elemen yang saat ini ada dalam himpunan untuk mendapatkan lebih banyak objek yang dapat dimasukkan ke dalam himpunan.
3. Bagian penutup: Bagian ini menyatakan bahwa satu-satunya elemen dalam himpunan adalah yang diperoleh dari (1) dan (2).

Contoh 1.3.5

Untuk mendefinisikan himpunan A bilangan bulat positif yang habis dibagi angka 5 secara rekursif, kita memiliki definisi rekursif yang terdiri dari tiga bagian berikut:

(a) 5 adalah elemen dari A.

(b) Jika n elemen A, maka n + 5 juga elemen A.

(c) Sebuah objek berada di A jika dan hanya jika objek tersebut diperoleh dengan penerapan berulang dari (a) dan (b).

Definisi Fungsi Rekursif

Misalkan (1) setiap elemen i dalam himpunan S = {0, 1, 2, . . . , k} dari k + 1 bilangan bulat pertama berturut-turut diberi bilangan real ri, dan (2) jika n adalah bilangan bulat yang lebih besar dari k, ada aturan f untuk mendefinisikan bilangan real f(n) yang dapat dinyatakan unik dalam hal beberapa atau semua istilah dari himpunan {f(n – 1), f(n – 2), . . . , f(n – k), f(n – k – 1)}. Jika kita sekarang mendefinisikan f(i) = ri untuk setiap i dalam S, aturan f adalah fungsi yang domainnya adalah himpunan bilangan bulat nonnegatif. Sebuah fungsi yang didefinisikan oleh metode ini disebut fungsi yang didefinisikan secara rekursif. Aturan yang mendefinisikan f(n) dalam kaitannya dengan nilai-nilai sebelumnya f(i) disebut relasi perulangan. Nilai f(0), f(1), f(2), . . . , f(k) disebut nilai awal dari relasi perulangan.

Kami menggunakan prinsip induksi (bentuk kuat) untuk menunjukkan bahwa definisi ini tidak melanggar definisi sebenarnya dari suatu fungsi, [yaitu, f(n) unik untuk setiap bilangan bulat nonnegatif n].

TEOREMA 1.3.1

Jika f didefinisikan secara rekursif, maka f(n) unik untuk setiap bilangan bulat positif n.

Bukti:

P(n) adalah proposisi bahwa f(n) unik untuk setiap n.

Langkah dasar:

Kami berasumsi bahwa f(i) = ri adalah unik ketika i = 0, 1, 2, . . . , k. Jadi P(0), P(1), . . . , P(k) benar.

Langkah induksi:

f(k + 1) dinyatakan secara unik dalam bentuk bilangan k + 1 ini. Jadi P(k+1) benar.

Contoh 1.3.6

Misalkan N adalah himpunan 11 bilangan bulat tak negatif dan R adalah himpunan semua bilangan real.

(a) Fungsi f: N → R yang mendefinisikan barisan $f(n) = 3^n$ dapat didefinisikan secara rekursif sebagai f(0) = 1 dan f(n) = 3f(n – 1) ketika n > 0.

(b) Fungsi (faktorial) f: N → R, di mana f(n) = n!, dapat didefinisikan secara rekursif sebagai f(0) = 1 dan f(n) = nf(n – 1) ketika n > 0 .

(c) Deret Fibonacci f: N → R didefinisikan secara rekursif oleh relasi f(n) = f(n – 1) + f(n – 2) ketika n > 1, dengan nilai awal f(0) = 0 dan f(1) = 1 memberikan barisan {0, 1, 1, 2, 3, 5, 8, 13, . . .}.

Penting bahwa dalam definisi rekursif, himpunan bilangan bulat pada langkah dasar yang merupakan kondisi awal adalah himpunan bilangan bulat berurutan. Jika tidak, fungsinya mungkin tidak terdefinisi dengan baik. Berikut adalah contoh tandingannya: f(n) = 9f(n – 2), dengan nilai awal yang tidak berurutan f(0) = 6 dan f(2) = 54, akan menghasilkan $f(n) = 2 \bullet 3^{n+1}$ juga sebagai $f(n) = 3 \bullet 3^n + 3 \bullet (-3)^n$.

1.4 BAHASA LOGIKA

Pada tingkat pengantar, logika matematika sangat mirip dengan teori himpunan. Alih-alih set, dalam logika kita memiliki proposisi. Proposisi adalah pernyataan yang bernilai benar atau salah tetapi tidak keduanya. Nilai kebenaran proposisi p adalah T (atau 1) jika p benar; jika tidak, nilai kebenarannya adalah F (atau 0). Perhatikan lima kalimat berikut:

1. p: 3 + 2 = 5
2. q: 3 + 2 = 6
3. r: Apakah 3 atau 2?
4. s: Ambil 3
5. t: x + 2 = 5

Di sini p adalah proposisi benar dan q adalah proposisi salah. Baik r dan s bukan proposisi, t adalah proposisi, tetapi tidak benar atau salah karena x tidak diketahui. Dalam teori himpunan kita memiliki perpotongan dan penyatuan dua himpunan dan komplemen dari suatu himpunan dalam himpunan universal tertentu. Konsep analog dalam logika adalah tiga operasi logis: konjungsi dua proposisi, disjungsi dua proposisi, dan negasi proposisi.

Konjungsi dua proposisi p dan q adalah proposisi yang benar jika dan hanya jika p dan q adalah proposisi yang benar. Konjungsi p dan q disebut “p dan q” dan dilambangkan dengan $p \wedge q$. Disjungsi dua proposisi p dan q adalah proposisi yang salah jika dan hanya jika p dan q adalah proposisi yang salah. Disjungsi p dan q disebut “p atau q” dan dilambangkan dengan $p \vee q$. Disjungsi eksklusif dua proposisi p dan q adalah proposisi yang benar jika dan hanya tepat satu dari keduanya benar dan dilambangkan dengan $p \oplus q$. Akhirnya, negasi dari proposisi p adalah proposisi p′ yang benar jika dan hanya jika p salah.

Sebuah tabel kebenaran menampilkan hubungan antara nilai-nilai kebenaran proposisi. Tabel berikut menampilkan nilai kebenaran konjungsi, disjungsi, disjungsi eksklusif, dan negasi dari dua proposisi p dan q:

Tabel 1.1 Hubungan antara nilai kebenaran proposisi

p	q	Conjunction	Disjunction	Exclusive disjunction	Negation of p
T	T	T	T	F	F
T	F	F	T	T	F
F	T	F	T	T	T
F	F	F	F	F	T

Proposisi yang dapat diperoleh dengan menggabungkan proposisi lain disebut proposisi majemuk. Misalnya, jika p, q, dan r adalah tiga proposisi, maka "(p' dan q) atau (q)" adalah proposisi majemuk. Proposisi yang bukan merupakan gabungan dari proposisi lain disebut proposisi atomik.

Operasi Implikasi

Ada cara lain yang penting untuk membangun proposisi majemuk dari dua proposisi p dan q. Ini adalah proposisi implikasinya: "jika p, maka q" (atau "p menyiratkan q") yang dilambangkan dengan p → q. Kami mendefinisikan proposisi majemuk ini salah jika dan hanya jika p benar dan q salah. Dalam semua kasus lain, proposisi majemuk p → q benar. Dalam hal ini kita katakan bahwa proposisi p adalah syarat cukup untuk proposisi q dan q syarat perlu untuk p. Di sini p disebut hipotesis (atau anteseden atau premis) dan q disebut konsekuensi (atau kesimpulan). Jelasnya, proposisi majemuk p → q dan proposisi majemuk q → p keduanya tidak mungkin salah pada saat yang bersamaan. Jadi terlepas dari nilai kebenaran p dan q, selalu terjadi bahwa setidaknya salah satu dari dua proposisi majemuk ini p → q dan q → p selalu benar.

Perhatikan bahwa konsep matematika dari implikasi lebih umum daripada konsep implikasi mengenai pernyataan dalam bahasa yang kita gunakan untuk berkomunikasi dalam kehidupan kita sehari-hari. Dalam pengaturan matematika umum tidak ada hubungan sebab-akibat antara nilai kebenaran hipotesis dan nilai kebenaran kesimpulan. Misalnya, jika p adalah proposisi bahwa "hari ini hujan" dan q adalah proposisi bahwa "London adalah ibu kota Inggris", maka proposisi implikasi p → q benar terlepas dari apakah p benar atau tidak. Implikasi bahwa "jika hari ini hujan, maka 3 + 4 = 8" adalah proposisi yang benar jika hari ini tidak hujan. Di sisi lain, ketika saya membuat pernyataan bahwa "jika hujan seperti ini, saya tidak akan pergi memancing siang ini", pernyataan ini merupakan proposisi implikasi di mana ada hubungan kausal yang pasti antara hipotesis dan kesimpulan.

Penting juga untuk disebutkan dalam konteks ini bahwa proposisi implikasi dalam banyak bahasa pemrograman memiliki nilai kebenaran yang berbeda. Jika sebuah baris dalam sebuah program mengatakan "jika n < 30 maka S," maka ketika eksekusi program mencapai baris ini, segmen S dijalankan jika n < 30 dan tidak dieksekusi sebaliknya.

Proposisi majemuk "p jika dan hanya jika q", yang dilambangkan dengan p q, adalah konjungsi antara proposisi majemuk p → q dan proposisi majemuk q → p. Proposisi majemuk p q benar jika p → q dan q → p keduanya benar. Dalam hal ini kita katakan bahwa p perlu dan cukup untuk q, dan sebaliknya. Berikut adalah tabel kebenaran dari operasi implikasi yang melibatkan dua operasi p dan q:

Tabel 1.2 Tabel kebenaran dari operasi implikasi

p	q	$p \rightarrow q$	$q \rightarrow p$	$p \leftrightarrow q$
F	F	T	T	T
F	T	T	F	F
T	F	F	T	F
T	T	T	T	T

Jika r adalah proposisi p → q, proposisi q → p adalah kebalikan dari r, proposisi p' → q' adalah invers dari r, dan proposisi q' → p' adalah kontrapositif dari r. Jika dua proposisi p dan q sedemikian rupa sehingga p benar jika dan hanya jika q benar, kedua proposisi p dan q dikatakan ekivalen. Kami menulis p = q ketika p dan q setara. Dengan kata lain, dua proposisi dikatakan ekivalen jika memiliki nilai kebenaran yang sama. Misalnya, proposisi bahwa "Jane akan berusia 18 tahun pada tahun 1993" sama dengan proposisi bahwa "Jane lahir pada tahun 1975." Proposisi majemuk p → q ekuivalen dengan proposisi kontrapositifnya q' → p' karena keduanya memiliki nilai kebenaran yang sama, seperti terlihat dari tabel kebenaran berikut:

Tabel 1.3 Tabel kebenaran

p	q	p'	q'	$p \rightarrow q$	$q' \rightarrow p'$
T	T	F	F	T	T
F	T	T	F	T	T
T	F	F	T	F	F
F	F	T	T	T	T

Proposisi majemuk yang selalu benar terlepas dari nilai kebenaran proposisi komponennya disebut tautologi. Proposisi majemuk yang selalu salah disebut kontradiksi. Sebagai contoh sederhana, disjungsi proposisi p dan negasinya merupakan tautologi, sedangkan konjungsi p dan negasinya merupakan kontradiksi. Dari tabel di atas kita perhatikan bahwa proposisi p q adalah tautologi jika dan hanya jika p = q.

Hukum komutatif, asosiatif, dan distributif yang melibatkan operasi konjungsi dan disjungsi dapat dengan mudah diverifikasi dengan menyusun tabel kebenaran yang sesuai. Hukum-hukum ini adalah:

1. Hukum komutatif:

$$p \wedge q = q \wedge p \text{ dan } p \vee q = q \vee p$$

2. Hukum asosiatif:

$$p \wedge (q \wedge r) = (p \wedge q) \wedge r \text{ dan } p \vee (q \vee r) = (p \vee q) \vee r$$

3. Hukum distributif:

$$p \wedge (q \vee r) = (p \wedge q) (p \vee r) \text{ dan } P \vee (q \wedge r) = (p \vee q) \wedge (p \vee r)$$

Nilai kebenaran proposisi majemuk bergantung pada nilai kebenaran proposisi komponennya. Subyek membangun dan menyederhanakan proposisi majemuk yang dibangun dari proposisi lain dan memperoleh nilai kebenarannya disebut kalkulus proposisional. Kombinasi proposisi untuk menghasilkan proposisi baru ini memiliki kemiripan yang kuat dengan kombinasi himpunan untuk membentuk himpunan baru. Teorema berikut ini analog dengan hukum De Morgan tentang teori himpunan.

TEOREMA 1.4.1

(a) $(p \wedge q)' = (p') \vee (q')$.

(b) $(p \vee q)' = (p') \wedge (q')$.

Bukti:

Ini dibiarkan sebagai latihan.

Masalah Kepuasan dalam Logika

Tabel kebenaran proposisi majemuk dengan n proposisi atomik sebagai komponennya akan memiliki 2n baris. Masalah kepuasan dari proposisi majemuk p adalah masalah (1) menemukan apakah ada nilai kebenaran untuk komponen atom p sedemikian rupa sehingga p benar, dan (2) memperoleh proposisi atom yang benar dan proposisi atom yang salah jika mereka ada yang membuat proposisi majemuk benar. Satu-satunya prosedur yang diketahui untuk menguji kelayakan proposisi dengan n proposisi atomik adalah membangun tabel kebenaran yang menyebutkan semua 2n kemungkinan nilai kebenaran, dan ini memang tugas yang berat jika n besar.

1.5 CATATAN DAN REFERENSI

Penyelidikan sistematis teori himpunan dimulai dengan kontribusi Georg Cantor (1848–1918) pada abad kesembilan belas. Sebelum itu, teori himpunan dan lebih umum matematika nonnumerik tidak diselidiki secara formal terlepas dari kontribusi George Peacock (1791-1858), Augustus De Morgan (1806-1871), dan George Boole (1815-1864). Peacock dan De Morgan menggeneralisasi operasi aljabar biasa di luar bidang matematika numerik dan Boole memperluas dan memformalkan kontribusi mereka dalam karya mani berjudul "Investigasi Hukum Pemikiran" pada tahun 1854. Menurut filsuf dan matematikawan abad kedua puluh yang hebat Bertrand Russell (1872–1970), George Boole-lah yang "menemukan" matematika murni.

Lebih lanjut tentang sejarah dan perkembangan teori himpunan dapat ditemukan di Boyer (1968). Untuk pembahasan rinci teori himpunan, termasuk fungsi dan hubungan, buku-buku oleh Halmos (1960) dan Stoll (1963) sangat direkomendasikan. Teknik pembuktian dengan induksi secara eksplisit dinyatakan dan digunakan oleh Francesco Maurolycus pada abad keenam belas ketika ia membuktikan bahwa jumlah n bilangan bulat positif ganjil pertama adalah n^2. Namun, teknik ini sudah dikenal matematikawan sejak abad ketiga SM. Misalnya, dalam bukti Euclid bahwa ada bilangan prima yang tak terbatas, teknik ini digunakan secara implisit.

Pada abad ketujuh belas, baik Pascal dan Fermat menggunakan metode induksi secara ekstensif. Istilah induksi matematika diciptakan oleh De Morgan. Pada abad kesembilan belas, prinsip induksi diselidiki secara rinci oleh Gottlieb Frege (1848–1925), Giuseppe Peano (1858–1932), dan Richard Dedekind (1831–1916). Peran induksi dalam perkembangan formal matematika menjadi fokus utama banyak ahli logika matematika pada awal abad kedua puluh, dan dua nama yang layak disebut dalam hal ini adalah Bertrand Russell dan Thoralf Skolem (1887– 1963). Untuk survei yang menarik tentang topik induksi matematika, lihat artikel oleh Bussey (1917). Golovina dan Yaglom (1963), Poly a (1954), dan Sominskii (1963) adalah tiga referensi yang sangat baik di bidang ini. Artikel Henkin (1960) juga sangat direkomendasikan.

Asal-usul studi sistematis tentang penalaran logis dapat ditelusuri ke Aristoteles, yang hidup pada abad keempat SM. Namun, baru pada abad ketujuh belas, simbol digunakan dalam studi logika. Karya perintis dalam logika simbolik dilakukan oleh Gottfried Leibniz (1646-1716). Tidak ada perkembangan besar yang terjadi sampai George Boole menerbitkan karyanya yang luar biasa yang disebutkan sebelumnya. Sejak saat itu Bertrand Russell dan Alfred North Whitehead (1861–1947) memberikan kontribusi yang besar terhadap perkembangan logika dan bidang logika matematika muncul ketika penemuan paradoks tertentu menyebabkan pemeriksaan ekstensif tempat logika, pembuktian, dan teori himpunan dalam dasar-dasar matematika.

1.6 LATIHAN

1. Misalkan A = {3, 5, 7, 9}, B = {2, 3, 5, 6, 7}, dan C = {2, 4, 6, 8} semua himpunan bagian dari alam semesta X = {2, 3, 4, 5, 6, 7, 8, 9}. Temukan (a) gabungan A dan B, (b) perpotongan B dan C, (c) B – A, (d) A – B, (e) komplemen mutlak C′ dari himpunan C, (f) komplemen absolut dari X, (g) komplemen absolut dari himpunan kosong.
2. Misalkan A, B, C seperti pada Soal 0.1. Tentukan himpunan berikut: (a) (A B) – C, (b) perpotongan (A B)′ dan (B C)′, (c) (A C) – (C – A)′
3. Manakah dari himpunan berikut yang sama? (a) {a, b, c, c}, (b) {a, b, a, b, c}, (c) {a, b, b, c, d}
4. Manakah dari himpunan berikut yang sama?
   (a) {t : t adalah akar dari $x^2 - 6x + 8 = 0$}
   (b) {y : y adalah bilangan real pada selang tertutup [2, 3]}
   (c) {4, 2, 5, 4}
   (d) {4, 5, 7, 2} – {5, 7}
   (e) {q : q adalah jumlah sisi persegi panjang atau jumlah digit dalam bilangan bulat antara 11 dan 99}
5. Jika A = {3, 4} dan B = {p, q, r}, sebutkan semua elemen dari (a) A × A, (b) A × B, (c) B × A dan (d) B × B.
6. Misalkan A dan B seperti pada Soal 1.5. Sebutkan semua elemen dari (a) A × A × A dan (b) A × A × B.
7. Misalkan A dan B seperti pada Soal 1.5. Sebutkan semua elemen dari (a) A (B × A) dan (b) (A × A) (B × A).
8. Sebutkan semua himpunan dalam himpunan pangkat dari himpunan berikut: (a) {a, b}, (b) {a, b, c}, (c) {φ, 0, {0}}
9. Daftar semua partisi dari himpunan berikut: (a) {a}, (b) {a, b}, (c) {a, b, c}

10. Tentukan apakah setiap pernyataan berikut benar dalam kasus tiga himpunan sembarang P, Q, R.
    (a) Jika P adalah elemen dari Q dan jika Q adalah himpunan bagian dari R, maka P adalah elemen dari R.
    (b) Jika P adalah anggota Q dan jika Q adalah himpunan bagian dari R, maka P juga merupakan himpunan bagian dari R.
    (c) Jika P adalah himpunan bagian dari Q dan Q adalah elemen dari R, maka P adalah elemen dari R.
    (d) Jika P adalah himpunan bagian dari Q dan Q adalah anggota dari R, maka P adalah himpunan bagian dari R.
11. Buktikan pernyataan berikut yang melibatkan tiga himpunan sembarang P, Q, dan R.
    (a) $(P - Q) - R = P - (Q\ R)$
    (b) $(P - Q) - R = (P - R) - Q$
    (c) $(P - Q) - R = (P - R) - (Q - R)$
12. Dua himpunan A dan B sedemikian rupa sehingga penyatuan dan perpotongannya sama. Apa yang bisa kita katakan tentang A dan B?
13. Misalkan A adalah himpunan bagian dari B dan C adalah himpunan bagian dari D.
    (a) Benarkah (A C) adalah himpunan bagian dari (B D)?
    (b) Benarkah perpotongan A dan C merupakan himpunan bagian dari perpotongan B dan D?
14. Apa yang dapat kita katakan tentang dua himpunan P dan Q jika $P - Q$ sama dengan $Q - P$?
15. Buktikan bahwa A dan B adalah himpunan tak kosong sehingga jika $A \times B$ dan $B \times A$ sama, maka $A = B$.
16. Berapa kardinalitas $P \times Q$ jika kardinalitas P adalah p dan kardinalitas Q adalah q?
17. Buktikan bahwa perpotongan himpunan pangkat A dan himpunan pangkat B adalah himpunan pangkat dari perpotongan A dan B, di mana A dan B adalah dua himpunan sembarang.
18. Apa yang dapat kita katakan jika operasi "persimpangan" diganti dengan operasi "serikat" dalam Soal 0.17?
19. Berapa kardinalitas himpunan daya dari himpunan kosong?
20. Jika powerset A sama dengan powerset B, apakah berarti A dan B sama?
21. Beda simetris dua himpunan A dan B adalah himpunan yang memuat elemen-elemen A atau B tetapi tidak keduanya A dan B dan dinotasikan dengan $A \oplus B$. Buktikan: $A \oplus B = (A - B) \cup (B - A) = (A \cup B) - (A \cap B)$.
22. Gambarlah diagram Venn untuk menunjukkan perbedaan simetris dari dua himpunan.
23. Berapa banyak daerah berbeda dalam diagram Venn yang mewakili tiga himpunan dalam himpunan semesta sehingga tidak ada perpotongan yang kosong?
24. Jika beda simetris dua himpunan A dan B sama dengan himpunan A, apa yang dapat kita katakan tentang A dan B?
25. Jika A dan B adalah dua himpunan sembarang, dalam kondisi apa kita dapat menyimpulkan bahwa beda simetris (A – B) dan (B – A) adalah himpunan kosong?

26. Dengan menggunakan diagram Venn, selidiki apakah pernyataan berikut ini benar atau salah.
    (a) $A \oplus (B \cap C) = (A \oplus B) \cap (A \oplus C)$
    (b) $A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C)$
    (c) $A \oplus (B \oplus C) = (A \oplus B) \oplus C$
    (d) $A \oplus (B \oplus C) = (A \cap B) \oplus (A \cap C)$
    (e) $A \cup (B \oplus C) = (A \cup B) \oplus (A \cup C)$
27. Jika beda simetri A dan B sama dengan beda simetris A dan C, apakah B = C perlu?
28. Buktikan pernyataan berikut yang melibatkan tiga himpunan sembarang A, B, dan C:
    (a) $A \times (B \cap C) = (A \times B) \cap (A \times C)$
    (b) $A \times (B \cup C) = (A \times B) \cup (A \times C)$
    (c) $(A \cap B) \times C = (A \times C) \cap (B \times C)$
    (d) $(A \cup B) \times C = (A \times C) \cup (B \times C)$
29. Misalkan R himpunan semua bilangan real dan f: R → R didefinisikan oleh $f(x) = x^2$.
    (a) Apa domain, kodomain, dan range dari fungsi ini?
    (b) Apakah f adalah injeksi?
    (c) Apakah f suatu surjeksi?
    (d) Temukan himpunan semua pragambar dari 4.
    (e) Tentukan bayangan terbalik dari himpunan $\{t : 1 \leq t \leq 4\}$.
30. Jika R adalah himpunan semua bilangan real, jelaskan mengapa F(x) = 1/(x – 2) dan F(x) = (akar kuadrat dari x) bukan fungsi dari R ke R.
31. Jika N adalah himpunan semua bilangan asli dan jika f: N → N didefinisikan oleh f(n) = 2n + 5, tunjukkan bahwa f adalah injeksi dan temukan fungsi inversnya. Apakah f adalah surjeksi? Apakah fungsi invers merupakan surjeksi?
32. Misalkan $f(x) = x^2 - 4$, di mana x adalah bilangan real. Tentukan bayangan dari himpunan berikut: (a) {–4, 4, 5}, (b) {4, 5} (c) {t : t adalah bilangan real yang lebih besar atau sama dengan nol}.
33. Misalkan A = {a, b, c, d} dan B = {p, q, r}.
    (a) Tentukan banyaknya fungsi dari A ke B.
    (b) Tentukan jumlah suntikan dari A ke B.
    (c) Tentukan banyaknya surjeksi dari A ke B.
    (d) Tentukan banyaknya fungsi sehingga a dipetakan ke p dan b dipetakan ke q.
34. Jika N adalah himpunan semua bilangan asli, berikan contoh fungsi dari N ke N yaitu (a) injeksi tetapi bukan surjeksi, (b) surjeksi tetapi bukan injeksi.
35. Temukan domain dan jangkauan fungsi yang memberikan (a) setiap bilangan bulat digit terakhirnya, (b) setiap bilangan bulat jumlah digit di dalamnya.
36. Berikan contoh fungsi f dari himpunan bilangan real ke himpunan bilangan real sehingga (a) f adalah injeksi sekaligus surjeksi, (b) f bukan injeksi maupun surjeksi.
37. Misalkan X = {p, q, r}, Y = {a, b, c, d}, dan Z = {1, 2, 3, 4}. Misalkan g: X → Y didefinisikan oleh himpunan pasangan terurut {(p, a), (q, b), (r, c)} dan f: Y → Z didefinisikan oleh

himpunan pasangan terurut {(a , 1), (b, 1), (c, 2), (d, 3)}. Tulislah fungsi komposit f g sebagai himpunan pasangan terurut.

38. Jika A = {p, q, r} dan f: A → A didefinisikan oleh f(p) = q, f(q) = p, dan f(r) = q, jelaskan f dan f f sebagai himpunan dari pasangan yang dipesan.
39. Misalkan A dan f seperti pada Soal 0.38. Tentukan $f^n = f \circ f \circ f \circ \cdots \circ f$ sebagai komposisi lipatan-n dari f dengan dirinya sendiri. Jelaskan f^n sebagai himpunan pasangan terurut ketika n ganjil dan n genap.
40. Tunjukkan bahwa himpunan semua bilangan bulat positif ekuivalen dengan himpunan semua bilangan bulat genap positif.
41. Misal f: B → C dan g: A → B. Buktikan:
    (a) Jika f dan g adalah injeksi, maka $f \circ$ g adalah injeksi.
    (b) Jika f dan g adalah surjeksi, maka $f \circ$ g adalah surjeksi.
42. Misalkan f dan g seperti pada Soal 0.41.
    (a) Misalkan $f \circ$ g adalah injeksi. Apakah perlu f menjadi suntikan? Apakah g perlu disuntik?
    (b) Misalkan $f \circ$ g adalah surjeksi. Apakah perlu f menjadi surjeksi? Apakah perlu g menjadi surjeksi?
43. Jika f(x) = ax + b dan g(x) = cx + d dan $f \circ g = g \circ f$, tentukan persamaan yang menghubungkan a, b, c, dan d.
44. Misalkan f: X → Y dan A dan B adalah himpunan bagian dari X. Maka buktikan: (a) f(A ⋃ B) = f(A) ⋃ f(B) dan (b) f(A ⋂ B) adalah a himpunan bagian dari perpotongan f(A) dan f(B).
45. Tunjukkan bahwa jika f: X → Y adalah injeksi, maka f(A ⋂ B) = f(A) ⋂ (B) untuk semua himpunan bagian A dan B dari X.
46. Tunjukkan bahwa ada injeksi dari A ke B jika dan hanya jika ada surjeksi dari B ke A.
47. Misalkan f: A → B dimana A dan B adalah dua himpunan berhingga dengan kardinalitas yang sama. Buktikan bahwa f adalah injeksi jika dan hanya jika f adalah surjeksi.
48. Misalkan A adalah himpunan bagian dari himpunan semesta X. Fungsi karakteristik fA dari A adalah fungsi dari X ke himpunan {0, 1} sedemikian sehingga bayangan setiap elemen di A adalah 1 dan bayangan setiap elemen yang tidak ada di A adalah 0. Misalkan A dan B adalah dua himpunan bagian dari X. Buktikan berikut untuk semua x dalam X.
    (a) $f_{A \cap B}(x) = f_A(x) \bullet f_B(x)$ untuk semua x dalam X
    (b) $f_{A \cup B}(x) = f_A(x) + f_B(x) - f_A(x) \bullet f_B(x)$
    (c) $f_A(x) + f_{A'}(x) = 1$
    (d) Jika C adalah selisih simetris A dan B, maka $f_C(x) = f_A(x) + f_B(x) - 2f_A(x) \bullet f_B(x)$.
49. Misalkan S = {0, 1} dan misalkan Sn adalah himpunan semua string dengan panjang di S. Jika u dan v adalah dua string di Sn, kita membandingkannya tempat demi tempat dan menentukan jarak Hamming H(u, v) antara u dan v menjadi jumlah tempat di mana mereka berbeda. Hitung jarak Hamming antara u dan v jika (a) u = 101100 dan v = 111011 (b) u = 01010 dan v = 11001.
50. Misalkan S, S_n, dan H(u, v) seperti pada Soal 0.49. Fungsi H: $S_n \times S_n$ → N (di mana N adalah himpunan semua bilangan bulat tak negatif) yang memetakan pasangan terurut

(u, v) ke dalam H(u, v) adalah fungsi jarak Hamming. Tunjukkan bahwa untuk semua u, v, dan w dalam S_n, fungsi H memenuhi aksioma metrik berikut: (a) H(u, v) nonnegatif, (b) H(u, v) = 0 jika dan hanya jika u = v, (c) H(u, v) = H(v, u), dan (d) H(u, v) H(u, w) + H(w, v).

51. Misalkan A = {1, 2, 3, 4, 5}, B = {a, b, c, d}, dan R adalah relasi {(1, a), (1, b), (3, c) , (4, d), (5, d), (5, c)}. Gambarkan hubungan ini dengan grafik bipartit dan gambar panah yang sesuai.
52. Berikan contoh relasi pada himpunan yang (a) simetris dan antisimetris, (b) tidak simetris dan antisimetris.
53. Misalkan A = {a, b, c, d}. Gambarlah diagram yang bersesuaian dengan masing-masing relasi berikut di A dan tentukan apakah setiap relasi refleksif, simetris, transitif, dan antisimetris. Periksa apakah properti perbandingan berlaku di salah satu hubungan ini.
    (a) R = {(b, b), (b, c), (b, d), (c, b), (c, c), (c, d)}
    (b) R = {(a, b), (b, a)}
    (c) R = {(a, a), (b, b), (c, c), (d, d)}
    (d) R = {(a, a), (b, b), (c, c), (d, d), (a, b), (b, a)}
    (e) R = {(a, c), (a, d), (b, c), (b, d), (c, a), (c, d)}
    (f) R = {(a, b), (b, c), (c, d)}
54. Misalkan R adalah relasi dari A ke B dan S adalah relasi dari B ke C. Maka relasi komposit S ∘ R dari R dan S adalah relasi yang terdiri dari semua pasangan terurut dari bentuk (a, c), di mana (a , b) ada di R dan (b, c) ada di S. Jika A = {p, q, r, s}, B = {a, b}, C = {1, 2, 3, 4}, R = {(p, a), (p, b), (q, b), (r, a) (s, a)} dan S = {(a, 1), (a, 2), (b, 4)}, cari S R.
55. Misalkan R suatu relasi pada himpunan A. Relasi R^2 pada A didefinisikan sebagai R ∘ R. Tunjukkan bahwa $R^2 \circ R$ sama dengan $R \circ R^2$. Jadi R^3 adalah gabungan dari R^2 dan R. Secara umum, pangkat ke-n R^n dari relasi tersebut adalah gabungan dari R^{n-1} dan R. Jika R = {(a, a), (a, b), (b, a), (c, b), (c, d)}, tentukan pangkat kedua dan ketiga dari R.
56. Buktikan bahwa jika suatu relasi pada suatu himpunan bersifat refleksif, maka sembarang pangkat dari relasi tersebut bersifat refleksif.
57. Buktikan bahwa jika suatu relasi R pada suatu himpunan bersifat refleksif dan transitif, maka $R^n = R$ untuk semua bilangan bulat positif n.
58. Misalkan R suatu relasi dari A ke B. Relasi invers R^{-1} dari B ke A adalah himpunan semua pasangan terurut dari bentuk (b, a), di mana (a, b) ada di R. Tunjukkan bahwa suatu relasi R pada suatu himpunan simetris jika dan hanya jika R dan inversnya sama.
59. Tunjukkan bahwa suatu relasi pada suatu himpunan adalah refleksif jika dan hanya jika relasi inversnya refleksif.
60. Buktikan bahwa suatu relasi R pada himpunan A adalah antisimetris jika dan hanya jika perpotongan dari R dan inversnya adalah himpunan bagian dari hubungan diagonal D = {(x, x): x ∈ A}.
61. Misalkan R adalah relasi pada himpunan A = {1, 2, 3, 4, 5, 6, 7} yang didefinisikan oleh aturan (a, b) ∈ R jika bilangan bulat (a – b) habis dibagi 4. Sebutkan elemen R dan inversnya.

62. Misalkan R adalah relasi pada himpunan N dari semua bilangan bulat positif yang didefinisikan oleh (a, b) ∈ R jika b habis dibagi a. Tentukan apakah R refleksif, simetris, antisimetris, atau transitif.
63. Misalkan N himpunan semua bilangan bulat positif dan misalkan R adalah relasi pada N × N yang didefinisikan oleh ((a, b), (c, d)) ada di R jika a $\leq$ c dan b $\leq$ d. Tentukan apakah R refleksif, simetris, antisimetris, atau transitif.
64. Manakah dari relasi berikut pada himpunan {1, 2, 3, 4} yang merupakan relasi ekivalen? Jika relasi tersebut merupakan relasi ekivalensi, buatlah daftar partisi yang sesuai (kelas ekivalensi).
    (a) {(1, 1), (2, 2), (3, 3), (4, 4), (1, 3), (3, 1)}
    (b) {(1, 0), (2, 2), (3, 3), (4, 4)}
    (c) {(1, 1), (2, 2), (1, 2), (2, 1), (3, 3), (4, 4)}
65. Misalkan R = {(x, y) : x dan y bilangan real dan x – y bilangan bulat}. Tunjukkan bahwa R adalah relasi ekivalen pada himpunan bilangan real.
66. Misalkan a bilangan bulat dan m bilangan bulat positif. Kami menyatakan dengan a (mod m) sisa ketika a dibagi dengan m. Jika a dan b dua bilangan bulat, kita katakan bahwa a kongruen dengan b modulo m jika m membagi a – b. Notasi a b (mod m) digunakan untuk menunjukkan bahwa a kongruen dengan b modulo m. Tentu saja, jika a kongruen dengan b modulo m, maka b kongruen dengan modulo m. Buktikan bahwa:

(a) a (mod m) = b (mod m) jika dan hanya jika a $\equiv$ b (mod m).

(b) a $\equiv$ b (mod m) jika dan hanya jika terdapat bilangan bulat k sedemikian rupa sehingga a = b + km.

(c) Jika a $\equiv$ b (mod m) dan c $\equiv$ d (mod m), maka a + c $\equiv$ (b + d) (mod m) dan ac $\equiv$ bd (mod m).

67. Misalkan Z himpunan semua bilangan bulat dan misalkan m bilangan bulat positif yang lebih besar dari 1. Tunjukkan bahwa relasi R pada Z yang didefinisikan oleh himpunan {(a, b) : a $\equiv$ b (mod m)} adalah relasi ekivalen. Relasi ini disebut relasi modulo m kongruensi pada himpunan bilangan bulat. Kelas ekivalen dari relasi ini disebut kelas kongruensi modulo m. Kelas kongruensi suatu bilangan bulat x modulo m dilekuk oleh $[x]_m$.
68. Carilah kelas-kelas yang kongruen modulo 5: (a) $[0]_5$, (b) $[1]_5$, dan (c) $[2]_5$.
69. Buktikan bahwa $\{[i]_m : i = 0, 1, 2, . . . , (m – 1)\}$ adalah partisi dari himpunan bilangan bulat.
70. Misalkan f adalah fungsi dari A ke A. Misalkan R adalah relasi pada A yang didefinisikan oleh {(x, y) : f(x) = f(y)}. Buktikan bahwa R adalah relasi ekivalensi pada A. Berapa kelas ekivalensinya?
71. Misalkan R adalah relasi ekivalensi pada himpunan tak kosong A. Tunjukkan bahwa ada fungsi f dengan A sebagai domain sedemikian sehingga (x, y) ada di R jika dan hanya jika f(x) = f(y).
72. Misalkan {(a, b), (c, d)} ada di R setiap kali a, b, c, d adalah bilangan bulat positif dan ad = bc. Tunjukkan bahwa R adalah relasi ekivalensi pada himpunan bilangan bulat positif.

73. Misal X = {1, 2, 3, 4, 5, . . . , 15}. Misalkan R adalah relasi pada X yang didefinisikan oleh (x, y) $\in$ R jika (x − y) habis dibagi 3. Buktikan bahwa R adalah relasi ekivalen pada X. Carilah kelas ekivalensi.
74. Misalkan R adalah relasi transitif dan refleksif pada himpunan A. Jika S adalah relasi pada A sedemikian sehingga (x, y) berada di S jika dan hanya jika (x, y) dan (y, x) keduanya berada di R, buktikan bahwa S adalah relasi ekivalensi pada A.
75. Buktikan bahwa relasi refleksif R pada himpunan A adalah relasi ekivalensi pada A jika dan hanya jika (x, y) dan (x, z) pada R menyiratkan bahwa (y, z) berada pada R.
76. Jika S adalah relasi yang didefinisikan pada himpunan bilangan real sebagai S = {(x, y) : $x \leq y$}, tunjukkan bahwa S adalah orde parsial pada himpunan bilangan real.
77. Jika S adalah relasi yang didefinisikan pada himpunan bilangan bulat positif sebagai S = {(x, y) : x membagi y}, tunjukkan bahwa S adalah orde parsial pada himpunan bilangan bulat positif.
78. Misalkan X = {a, b, c} dan S adalah orde parsial yang didefinisikan pada himpunan daya P(X) yang didefinisikan sebagai S = {(A, B) : A adalah himpunan bagian dari B}. Sebutkan unsur-unsur S.
79. Gambarlah diagram Hasse dari orde parsial S dari Soal 0.78.
80. Misalkan X = {1, 2, 3, 4, 5, 6, 7, 8, 9} dan S = {(m, n), di mana m membagi n} adalah orde parsial pada X. Gambarlah diagram Hasse yang mewakili S. Cari rantai di X.
81. Buktikan bahwa jika R adalah orde parsial pada himpunan A, inversnya R^{-1} juga merupakan orde parsial pada A. Himpunan terurut sebagian (A, R^{-1}) disebut dual dari himpunan terurut sebagian (A, R).
82. Misalkan A = {2, 3, 4, 6, 8, 12, 16, 24} dan S adalah relasi orde parsial pada A yang didefinisikan oleh S = {(a, b) : a membagi b}. Temukan (a) elemen minimal di A, (b) elemen maksimal di A, dan (c) batas atas himpunan B = {4, 6, 12}.
83. Gambarlah diagram Hasse dari himpunan PO pada Soal 0.82.
84. Buktikan bahwa (a) setiap himpunan terurut parsial hingga memiliki elemen maksimal dan elemen minimal, dan (b) setiap himpunan terurut linier hingga memiliki elemen terbesar dan elemen terkecil.
85. Buktikan dengan induksi bahwa $1^k + 2^k + 3^k + \cdots +$ nk sama dengan (a) n(n + 1) (2n + 1)/6 ketika k = 2, dan (b) [n(n + 1) /2]2 ketika k = 3.
86. Buktikan dengan induksi bahwa 1.2 + 2.3 + 3.4 + ⋯ + n(n + 1) sama dengan n(n + 1)(n + 2)/3.
87. Buktikan dengan induksi bahwa 1/1.2 + 1/2.3 + 1/3.4 + ⋯+ 1/n(n + 1) sama dengan n/(n + 1).
88. Tunjukkan bahwa $n^3 + 2n$ habis dibagi 3 untuk semua bilangan bulat positif n.
89. Buktikan bahwa 1.2.3 + 2.3.4 + 3.4.5 + ⋯+ n(n + 1)(n + 2) sama dengan n(n + 1)(n + 2)(n + 3)/4 .
90. Buktikan bahwa 1^2/1.3 + 22/3.5 + ⋯ + n^2/(2n – 1)(2n + 1) sama dengan [n(n + 1)]/2(2n + 1).
91. Tunjukkan bahwa jumlah pangkat tiga dari tiga bilangan bulat positif berurutan habis dibagi 9.

92. Tunjukkan bahwa untuk sembarang bilangan bulat positif n yang lebih besar dari 1, jumlah $1 + 1/\sqrt{2} + 1/\sqrt{3} + \cdots + 1/\sqrt{n}$ lebih besar dari $\sqrt{n}$.
93. Buktikan: (1 – 1/2)(1 – 1/3) ⋯ (1 – 1/n) = 1/n.
94. Buktikan: $2^n > n^2$ bila n > 4.
95. Tunjukkan bahwa 1/(n + 1) + 1/(n + 2) + ⋯ + 1/(2n) lebih besar dari 13/24 setiap kali n lebih besar dari 1.
96. Buktikan bahwa $7^n – 1$ habis dibagi 6.
97. Buktikan bahwa $11^n – 6$ habis dibagi 5.
98. Tunjukkan bahwa $6,7^n – 2,3^n$ habis dibagi 4.
99. Buktikan: $3^n + 7^n – 2$ habis dibagi 8.
100. Buktikan hukum De Morgan:
    (a) Komplemen mutlak dari perpotongan n himpunan bagian dari suatu himpunan semesta sama dengan gabungan dari komplemen mutlak dari n himpunan tersebut.
    (b) Komplemen absolut dari gabungan n himpunan ini adalah perpotongan dari komplemen absolutnya.
101. Tunjukkan bahwa kardinalitas himpunan pangkat dari suatu himpunan dengan n elemen adalah 2^n.
102. Buktikan bahwa jika S adalah relasi transitif pada himpunan A, maka S^n adalah himpunan bagian dari S untuk n = 1, 2, 3, ...
103. Misalkan f didefinisikan secara rekursif sebagai f(0) = 1 dan f(n + 1) = 3f(n) + 5. Carilah f(1), f(2), dan f(3).
104. Berikan definisi rekursif dari xn ketika x adalah bilangan real dan n adalah bilangan bulat nonnegatif.
105. Berikan definisi rekursif dari f di mana f(n) adalah jumlah dari n bilangan bulat positif pertama.
106. Berikan definisi rekursif dari himpunan (a) semua bilangan bulat, (b) semua bilangan bulat ganjil positif, (c) semua bilangan bulat genap negatif dan (d) semua bilangan bulat genap.
107. Buatlah tabel kebenaran dari pernyataan p → (p ∨ q) dan tentukan apakah itu tautologi atau kontradiksi atau bukan keduanya.
108. Tunjukkan bahwa (p′ ∨ q) ∧ (p ∧ (p ∧ q)) setara dengan (p ∧ q).
109. Periksa apakah (p → q) → r merupakan tautologi atau kontradiksi.
110. Buatlah tabel kebenaran dari pernyataan [(p → q) ∧ (q → p)] ⟷ (p ⟷ q) dan tentukan apakah itu kontradiksi.
111. Buatlah tabel kebenaran dari q ⟷ (p′ ∨ q′).
112. Misalkan p dan r adalah pernyataan yang salah dan q dan s adalah pernyataan yang benar. Tentukan nilai kebenaran dari
    (a) (p → q) → r
    (b) (s → (p ∧ r′)) ∧ ((p → (r ∨ q)) ∧ s)
113. Tentukan penugasan kebenaran dari p, q, r, s, dan t sedemikian sehingga memenuhi:
    (a) (p ∧ q ∧ r) → (s ∨ t)
    (b) (p′ ∧ q) ∨ r′

114. Tunjukkan bahwa $(p \rightarrow q) \rightarrow (p' \vee q)$ adalah tautologi.
115. Buktikan: $(p \wedge q) \wedge (p \vee q)'$ adalah kontradiksi.
116. Tunjukkan bahwa $((p \rightarrow q) \rightarrow q)$ $((p - q)' \vee q)$ adalah tautologi.

BAB 2
KOMBINATORIK

2.1 DUA ATURAN PENGHITUNGAN DASAR

Kombinatorika adalah salah satu bidang matematika modern yang tumbuh paling cepat. Ini memiliki banyak aplikasi untuk beberapa bidang matematika dan terutama berkaitan dengan studi himpunan hingga atau diskrit (seperti himpunan bilangan bulat) dan berbagai struktur pada himpunan ini, seperti pengaturan, kombinasi, tugas, dan konfigurasi. Secara umum, tiga jenis masalah muncul saat mempelajari himpunan dan struktur ini: (1) masalah keberadaan, (2) masalah penghitungan, dan (3) masalah optimasi. Masalah keberadaan berkaitan dengan pertanyaan berikut: Apakah ada setidaknya satu pengaturan dari jenis tertentu? Masalah penghitungan, di sisi lain, berusaha menemukan jumlah kemungkinan pengaturan atau konfigurasi dari pola tertentu. Masalah menemukan pengaturan yang paling efisien dari pola yang diberikan adalah masalah optimasi. Dalam bab ini kita mempelajari teknik untuk memecahkan masalah yang melibatkan penghitungan. Teknik-teknik ini membentuk dasar untuk studi kombinatorik enumeratif, yang sebenarnya merupakan teori penghitungan di mana hasil yang melibatkan penghitungan diperoleh tanpa melakukan proses penghitungan yang tepat, yang bisa jadi membosankan.

Misalkan ada 10 jurusan matematika dan 15 jurusan ilmu komputer dalam sebuah kelas yang terdiri dari 25 dan kita diminta untuk memilih seorang siswa dari kelas tersebut untuk mewakili matematika dan seorang siswa lainnya untuk mewakili ilmu komputer. Sekarang ada 10 cara memilih jurusan matematika dan 15 cara memilih jurusan ilmu komputer dari kelas tersebut. Selanjutnya, tindakan memilih siswa dari satu bidang sama sekali tidak tergantung pada tindakan memilih siswa dari yang lain. Jadi secara intuitif jelas bahwa ada $10 \times 15 = 150$ cara untuk memilih perwakilan dari matematika dan perwakilan dari ilmu komputer. Di sisi lain, jika kita diminta untuk memilih satu perwakilan dari matematika atau dari ilmu komputer, kita hanya memiliki $10 + 15 = 25$ cara untuk menyelesaikannya. Dalam kasus pertama kami menggunakan aturan perkalian penghitungan dan yang terakhir aturan penambahan. Kedua aturan tersebut dapat dinyatakan secara formal sebagai berikut.

Aturan Multiplikasi (Aturan Penghitungan Berurutan)

Misalkan ada barisan dari r kejadian $E_1, E_2, \ldots, E_r$ sedemikian rupa sehingga (1) ada ni cara di mana Ei(i = 1, 2, . . , r) dapat terjadi, dan (2) banyaknya cara suatu peristiwa dalam barisan dapat terjadi tidak bergantung pada bagaimana peristiwa dalam urutan sebelum peristiwa itu terjadi. Lalu ada $(n_1) \bullet (n_2) \bullet \ldots \bullet (n_r)$ cara-cara di mana semua peristiwa dalam urutan dapat terjadi.

Aturan Tambahan (Aturan Penghitungan Disjungtif)

Misalkan ada r kejadian $E_1, E_2, \ldots$, Er sedemikian rupa sehingga (1) terdapat ni hasil untuk Ei(i = 1, 2, . . , r), dan (2) tidak ada dua kejadian yang dapat terjadi secara bersamaan. Maka ada $(n_1) + (n_2) + \bullet\bullet\bullet + (n_r)$ cara di mana salah satu dari r peristiwa ini dapat terjadi.

Kedua aturan dasar ini sangat berguna dalam menyelesaikan masalah berhitung tanpa melakukan pencacahan secara eksplisit. Namun, jika seseorang tidak berhati-hati, kemungkinan besar akan disalahgunakan, menghasilkan hasil yang salah, seperti yang dapat dilihat dari beberapa contoh yang akan kita bahas berikut ini.

Contoh 2.1.1

Ada lima karakter—dua huruf alfabet diikuti tiga angka—yang muncul di belakang salah satu rangkaian mikrokomputer yang dibuat oleh perusahaan elektronik. Jumlah komputer yang mungkin diproduksi di sini deret adalah (1) $26 \times 26 \times 10 \times 10 \times 10 = 676.000$ jika karakter dapat diulang, (2) $26 \times 25 \times 10 \times 10 \times 10 = 650.000$ jika huruf tidak dapat diulang, dan (3) $26 \times 25 \times 10 \times 9 \times 8 = 468.000$ jika tidak ada karakter yang dapat diulang. Kami menggunakan aturan perkalian di sini.

Contoh 2.1.2

Seorang profesor memiliki 25 siswa dalam kursus kalkulus lanjutannya dan 31 siswa dalam kursus statistiknya. Tiga belas siswa telah mendaftar untuk kedua kursus. Ada tiga peristiwa di sini, tidak ada dua yang dapat terjadi secara bersamaan: (1) Peristiwa bahwa seorang siswa yang dipilih secara acak telah mendaftar untuk kalkulus tingkat lanjut tetapi tidak untuk statistik, dan ini dapat terjadi dalam 12 cara; (2) peristiwa bahwa seorang siswa yang dipilih secara acak telah mendaftar untuk statistik tetapi tidak untuk kalkulus tingkat lanjut, dan ini dapat terjadi dalam 18 cara; dan (3) kejadian dimana seorang siswa yang dipilih secara acak telah mendaftar untuk kedua mata kuliah tersebut dan ini dapat terjadi dalam 13 cara. Dengan aturan penjumlahan salah satu kejadian ini dapat terjadi dalam $12 + 18 + 13 = 43$ cara.

Dengan kata lain, profesor memiliki 43 siswa di kedua mata kuliah tersebut secara bersamaan. Perhatikan bahwa peristiwa bahwa seorang siswa yang dipilih secara acak mengambil kalkulus tingkat lanjut dan peristiwa bahwa seorang siswa yang dipilih secara acak mengambil statistik dapat terjadi secara bersamaan. Jadi kita tidak dapat menerapkan aturan penjumlahan pada dua kejadian untuk menyimpulkan bahwa profesor memiliki total $25 + 31 = 56$ siswa.

Contoh 2.1.3

Dalam kelompok tamasya ada 8 orang Austria, 5 orang Brasil, dan 6 orang Kanada. Jadi dengan aturan perkalian ada 40 cara untuk memilih seorang Austria dan seorang Brazil, 48 cara untuk memilih seorang Austria dan seorang Kanada, dan 30 cara untuk memilih seorang Brazil dan seorang Kanada. Selanjutnya, dengan prinsip penjumlahan, terdapat $40 + 48 + 30 = 118$ cara untuk memilih sepasang individu yang berbeda kebangsaan dari kelompok wisatawan ini. Sebuah tim yang terdiri dari 3 turis dari kebangsaan yang berbeda dapat dipilih dalam $8 \times 5 \times 6$ cara, sedangkan perwakilan tipikal dapat dipilih dengan $8 + 5 + 6$ cara.

Contoh 2.1.4

Jumlah bilangan bulat ganjil antara 0 dan 99 jelas 50. Kita dapat menggunakan aturan perkalian untuk mendapatkan hasil ini. Setiap bilangan bulat antara 0 dan 99 memiliki digit satuan dan digit puluhan jika kita menulis 0, 1, 2, . . ., 9 sebagai 00, 01, 02, . . . , 09. Misalkan E adalah kejadian pemilihan suatu digit untuk digit satuan. Ini bisa dilakukan dengan 5 cara. Selanjutnya, misalkan F adalah kejadian pemilihan angka untuk angka puluhan. Ini bisa dilakukan dengan 10 cara. Perhatikan bahwa banyaknya cara E dapat terjadi tidak bergantung pada bagaimana F dapat terjadi, dan sebaliknya. Jadi barisan E, F (atau dalam hal ini, barisan F, E) dapat terjadi dalam 50 cara.

Contoh 2.1.5

Misalkan kita tertarik untuk menemukan jumlah bilangan bulat ganjil antara 0 dan 100 dengan angka yang berbeda. Misalkan E dan F seperti pada Contoh 1.1.4. E dapat dilakukan dengan 5 cara seperti sebelumnya. Setelah itu F dapat terjadi dalam 9 cara. Banyaknya cara F dapat terjadi tidak bergantung pada bagaimana E terjadi. Jadi dengan aturan perkalian barisan E, F dapat terjadi dalam 45 cara, dan akibatnya, ada 45 bilangan bulat seperti itu. Di sisi lain, jika F adalah peristiwa pertama, itu dapat terjadi dalam 10 cara. Selanjutnya, kejadian kedua E dapat dilakukan dengan 5 cara jika angka puluhannya genap, dan dengan 4 cara jika angka puluhannya ganjil. Dengan kata lain, banyaknya cara terjadinya E bergantung pada bagaimana peristiwa F terjadi. Jadi kita tidak bisa menerapkan aturan perkalian pada barisan F, E dalam kasus ini.

Contoh 2.1.6

Misalkan X adalah suatu himpunan dengan n anggota. Sebutkan unsur-unsur X sebagai 1, 2, . . . , n dan perhatikan urutan n peristiwa berikut: Peristiwa pertama adalah memutuskan apakah akan memilih elemen pertama atau tidak, peristiwa kedua adalah memutuskan apakah akan memilih elemen kedua atau tidak, dan seterusnya. Setiap peristiwa dapat terjadi dalam 2 cara dan banyaknya cara agar salah satu peristiwa dalam urutan ini dapat terjadi tidak bergantung pada bagaimana peristiwa sebelumnya dalam urutan itu terjadi. Jadi setiap himpunan dengan n elemen memiliki 2n himpunan bagian, dengan aturan perkalian. Kelas dari semua himpunan bagian dari himpunan X adalah himpunan pangkat dari X dan dilambangkan dengan P(X) seperti yang disebutkan dalam Bab 0.

2.2 PERMUTASI

Pertimbangkan koleksi X dari n objek berbeda. Permutasi-r dari X adalah suatu susunan dalam deretan r objek dari X. Tentu saja, r paling banyak adalah n. Jadi jika X adalah himpunan dari 5 huruf pertama a, b, c, d, dan e, maka edcb, dbea, dan bdca adalah beberapa dari 4 permutasi X. Banyaknya r-permutasi dari suatu himpunan dari n objek berbeda dilambangkan dengan P(n, r). Setiap permutasi r di sini dapat dianggap sebagai urutan r peristiwa di mana jumlah cara suatu peristiwa dapat terjadi tidak bergantung pada bagaimana peristiwa sebelum peristiwa itu terjadi. Jadi kita menggunakan aturan perkalian untuk menyimpulkan bahwa P(n, r) sama dengan $n(n - 1)(n - 2) \cdots (n - r + 1)$ karena sembarang objek dari X dapat dipilih dalam n cara dan setelah memilih itu, objek arbitrer kedua dapat dipilih dalam (n – 1) cara, dan seterusnya, sampai semua r objek dipilih.

Masalah Permutasi dan Alokasi

Kita dapat mendekati proses membuat pengaturan objek dari sudut pandang yang berbeda. Pertimbangkan satu set n lokasi berbeda yang diatur dalam urutan yang pasti dan kami diminta untuk mengalokasikan r objek berbeda ke lokasi ini sedemikian rupa sehingga tidak ada lokasi yang dapat menerima lebih dari satu objek. Maka banyaknya cara untuk mengalokasikan r objek ini ke n lokasi juga P(n, r) dengan aturan perkalian karena sembarang objek dapat dikirim ke salah satu lokasi dengan n cara, dan selanjutnya objek lain dapat dikirim dalam (n – 1) cara, dan seterusnya.

Contoh 2.2.1

Jika X = {1, 2, 3, 4, 5, 6, 7} dan r = 3, banyaknya r-permutasi dari X adalah $7 \times 6 \times 5 = 210$.

Setiap n-permutasi dari himpunan X dengan n elemen disebut permutasi dari X dan jumlah P(n, n) dari permutasi X adalah n(n – 1)(n – 2) • . . . • 3 • 2 • 1, yang dilambangkan dengan fungsi faktorial n!. Sangat mudah untuk melihat bahwa P(n, r) = n!/(n – r)!. (Kami mendefinisikan 0! = 1.)

Bilangan bulat positif n! bisa sangat besar bahkan ketika n adalah angka dua digit kecil. Ini lebih dari 3,6 juta ketika n = 10 dan kira-kira sama dengan (2.433)(10^{18}) ketika n = 20.

Permutasi Lingkaran dan Cincin

Contoh 2.2.2

Pertimbangkan koleksi 5 batu dengan warna berbeda: biru (B), hijau (G), merah (R), merah muda (P), dan putih (W).

(a) Banyaknya cara membuat peniti di mana 5 batu ini akan ditempatkan secara horizontal, tentu saja, 5!.

(b) Dalam berapa banyak cara kita dapat membuat peniti di mana batu-batu ini ditempatkan dalam pola melingkar? Jawabannya harus kurang dari 5! karena beberapa permutasi yang dipertimbangkan dalam (a) sekarang tidak berbeda. Misalnya, jika kita memutar permutasi BGRPW satu kali searah jarum jam, kita mendapatkan permutasi GRPWB, dan kedua permutasi ini tidak berbeda dalam susunan melingkar. Jika kita memperbaiki salah satu warna dan kemudian mempertimbangkan permutasi yang dibentuk oleh 4 warna yang tersisa, semua permutasi ini berbeda. Misalnya, jika kita memperbaiki B dan mempertimbangkan RGPW dan RGWP, kita mendapatkan dua permutasi, BRGPW dan BRGWP, yang berbeda. Jadi hanya ada (4!) permutasi melingkar seperti itu.

(c) Dengan cara bagaimana kita dapat membuat cincin yang di dalamnya batu-batu tersebut dipasang? Dalam sebuah cincin, tidak ada perbedaan antara permutasi dan "gambar cermin" -nya. Misalnya, BGRPW dan BWPRG adalah sama. Untuk setiap permutasi pada (b), ada bayangan cermin. Jadi jawabannya sekarang adalah (4!)/2.

Jadi banyaknya permutasi sirkular dari suatu himpunan yang terdiri dari n elemen adalah (n – 1)! dan banyaknya permutasi ring adalah ((n – 1)!)/2.

Permutasi Umum

Mari kita perhatikan koleksi X dari n objek (tidak harus berbeda) yang termasuk dalam k grup tak kosong yang berbeda sehingga (1) semua objek dalam suatu grup identik, dan (2) objek dalam grup tidak identik dengan objek dalam kelompok lain. (Misalnya, huruf-huruf pada himpunan a, b, a, b, b, d, e, e, d dapat dibentuk menjadi empat kelompok: satu untuk a, satu untuk b, satu untuk d, dan satu untuk e.) Asumsikan ada ni objek dalam kelompok i dimana i = 1, 2, . . ., k. Setiap susunan dalam deretan n objek ini disebut permutasi umum X. (Misalnya, LINISOIL adalah permutasi umum dari huruf-huruf yang muncul dalam kata ILLINOIS.) Jumlah permutasi umum tersebut dilambangkan dengan P(n; n_1, n_2, . . , n_k), yang akan menjadi n! jika semua objek di X berbeda.

TEOREMA 2.2.1

Jika kumpulan X dari n objek terdiri dari k grup tak kosong yang berbeda sehingga grup i memiliki ni objek identik (di mana i = 1, 2, . . ., k), maka banyaknya permutasi umum dari X adalah (n!)/(n_1!)(n_2!) ⋯ (n_k!).

Bukti:

Jika benda-benda milik kelompok i semuanya berbeda, pasti ada ni! permutasi untuk elemen-elemen dalam grup ini. Jadi setiap permutasi umum menghasilkan N = $(n_1!)(n_2!) \cdots (n_k!)$ permutasi dari X jika X memiliki objek yang berbeda. Jika t adalah jumlah total permutasi umum, kita memiliki (t)(N) = n!, dari mana kesimpulan teorema berikut. [Amati bahwa jika k = n, setiap grup memiliki tepat satu elemen yang ekuivalen dengan pernyataan bahwa objek di X berbeda dengan memverifikasi bahwa P(n; 1, 1, . . , 1), di mana 1 diulang n kali , sama dengan n!, sebagaimana mestinya.]

Contoh 2.2.3

9 huruf yang muncul pada kata KONSENSUS dapat dikelompokkan menjadi 6 kelompok: kelompok yang terdiri dari tiga huruf S, kelompok yang terdiri dari dua huruf N, dan empat kelompok yang masing-masing terdiri dari empat huruf berbeda yang tersisa. Jumlah total permutasi umum dalam kasus ini adalah (9!)/(3!)(2!)(1!)(1!)(1!)(1!) = 30.240.

Jika jumlah total objek dalam setiap (k – 1) dari kelompok k ini adalah r (di mana r n), rumus untuk jumlah permutasi umum dapat dinyatakan sebagai P(n, r)/$(n_1!)(n_2!) \cdots (n_{k-1}!)$ sejak n! = P(n, r) • (n – r)! dan nk = (n – r). Sebagai contoh:

$$P(15; 2, 3, 4, 6) = \frac{(15!)}{(2!)(3!)(4!)(6!)} = \frac{P(15, 9)}{(2!)(3!)(4!)} = \frac{P(15, 12)}{(2!)(4!)(6!)}$$

dan seterusnya. Jadi jika ni (i = 1, 2, . .., k) adalah k bilangan bulat positif yang jumlahnya r di mana r n dan jika kita definisikan:

$$P(n; n_1, n_2, \ . \ . \ . \ , n_k) = \frac{P(n, r)}{(n_1!)(n_2!) \cdot \cdot \cdot (n_k!)}$$

kita melihat itu:

1) P(n; n1, n2, . . . , nk –1) = P(n; n_1, n_2, . . , n_{k-1}, m) di mana m = n – ($n_1 + n_2 + \cdots + n_k - 1$)
2) P(n; r) = P(n; n – r) = P(n; r, n – r)
3) (r!) P(n; r) = P(n, r)

Contoh 2.2.4

$$P(15; 2, 4, 4) = P(15; 2, 4, 4, 5) = \frac{(15!)}{(2!)(4!)(4!)(5!)} = \frac{P(15, 10)}{(2!)(4!)(4!)}$$

Kami sekarang memiliki generalisasi Teorema 2.2.1 berikut, buktinya dibiarkan sebagai latihan sederhana.

TEOREMA 2.2.2

Jika terdapat ni objek identik dalam kelompok i (i = 1, 2, . . , k) dan jika r adalah jumlah total objek dalam kelompok k ini, r objek tersebut dapat ditempatkan di n lokasi berbeda, sehingga setiap lokasi menerima paling banyak satu objek, dalam t cara, di mana t = P(n; n_1, n_2, . . , n_k). Khususnya, jika setiap grup memiliki tepat satu objek, maka t = P(n, r), yang merupakan jumlah permutasi dari suatu himpunan dengan n elemen.

Contoh 2.2.5

Misalkan ada 100 tempat (ditandai secara berurutan dari 100 hingga 199) di ruang pamer sebuah dealer mobil untuk memajang mobil baru di mana 15 mobil sport, 25 mobil kompak, 30 station wagon, dan 20 van diparkir untuk dipamerkan. Asumsikan bahwa mobil di setiap kategori adalah merek baru dan identik dalam segala hal, termasuk warna. Dealer kemudian dapat memarkir koleksi 90 kendaraan untuk dipajang (meninggalkan 10 tempat kosong di tempat parkir) dengan cara P(100, 90)/(15!) (25!)(30!)(20!).

2.3 KOMBINASI

Seperti pada bagian 1.2, misalkan X adalah kumpulan dari n objek berbeda. Sekumpulan r objek berbeda dari X disebut kombinasi r dari X. Dengan kata lain, jika X adalah himpunan dengan n elemen, setiap himpunan bagian dari X dengan r elemen adalah kombinasi-r dari X. Dalam kombinasi-r urutan di mana elemen r dipilih tidak penting, tidak seperti dalam kasus permutasi r. Jumlah r-kombinasi dari suatu himpunan dengan n elemen dilambangkan dengan C(n, r), yang merupakan jumlah himpunan bagian dari kardinalitas r. Jadi terdapat P(n, 2) pasangan terurut dan C(n, 2) pasangan tak berurut dari dua elemen dalam himpunan n elemen. Tentu saja, C(n, 0) = C(n, n) = 1.

Apa hubungan antara C(n, r) dan P(n, r)? Pertimbangkan setiap himpunan bagian A dari X dengan r elemen. R elemen yang berbeda ini dapat diatur dengan cara (r!). Jadi ada (r!) permutasi yang terkait dengan setiap subset elemen-r dari X. Tentu saja, menurut definisi, jumlah subset elemen-r dari X adalah C(n, r). Jadi jumlah total r-permutasi adalah hasil kali (r!) dan C(n, r) dengan aturan perkalian. Jadi kita memiliki teorema penting berikut.

TEOREMA 2.3.1

$$C(n, r) \bullet (r!) = P(n, r)$$

Akibat Wajar

$$C(n, r) = C(n, n - r)$$

Bukti:

$$C(n, r) = \frac{(n!)}{r!\,(n-r)!} = \frac{(n!)}{(n-r)!\,r!}$$

$$C(n, n-r) = \frac{(n!)}{(n-r)!\,(n-(n-r))!} = \frac{(n!)}{(n-r)!\,r!}$$

Dengan kata lain, jika X adalah himpunan dengan n anggota, jumlah himpunan bagian dari X dengan r elemen sama dengan jumlah himpunan bagian dari X dengan (n – r) elemen.

Kombinasi dan Masalah Alokasi

Seperti dalam kasus permutasi, kita dapat menginterpretasikan kombinasi dari sudut pandang yang berbeda, sebagai masalah alokasi. Seperti sebelumnya, misalkan X adalah himpunan n lokasi berbeda yang disusun dalam urutan tertentu dan pertimbangkan kumpulan r objek yang identik. Objek-objek ini akan dialokasikan ke n lokasi ini sehingga tidak ada lokasi yang menerima lebih dari satu subjek. Misalkan t adalah jumlah total cara untuk mengalokasikan r objek ini. Jika semua objek berbeda, setiap alokasi tersebut akan menghasilkan alokasi (r!). Dalam hal ini jumlah total alokasi adalah (t)(r!). Tetapi jumlah total alokasi jika objeknya berbeda adalah P(n, r). Jadi t = P(n, r)/(r!) = C(n, r).

TEOREMA 2.3.2 (Rumus Pascal)

$$C(n, r) = C(n - 1, r) + C(n - 1, r - 1)$$

Bukti:

Misalkan X adalah himpunan dengan n elemen dan Y adalah himpunan bagian dari X dengan (n – 1) elemen. Misalkan t adalah elemen X yang tidak ada di Y. Setiap subset elemen r dari X adalah subset elemen r dari Y atau gabungan dari subset Y dengan (r – 1) elemen dan himpunan tunggal yang terdiri dari t. Pada kategori pertama terdapat himpunan C(n – 1, r) dan pada kategori kedua terdapat himpunan C(n – 1, r – 1). Dengan kata lain, jumlah himpunan bagian dari X dengan r elemen adalah jumlah dari C(n – 1, r) dan C(n – 1, r – 1).

Rumus Pascal adalah contoh identitas kombinatorial, yang dibuktikan dengan menggunakan argumen kombinatorial. Identitas ini juga dapat dibuktikan secara aljabar. Beberapa identitas kombinatorial diberikan di akhir bab ini sebagai latihan. Berikut adalah contoh lain.

Contoh 2.3.1

$$C(2n, 2) = 2C(n, 2) + n^2$$

Bukti:

Misalkan X adalah sembarang himpunan dengan 2n elemen yang dipartisi menjadi dua himpunan Y dan Z, masing-masing berisi n elemen. Banyaknya himpunan bagian dari X dengan dua elemen adalah C(2n, 2). Setiap himpunan bagian dari X memiliki dua elemen jika dan hanya jika itu milik salah satu dari tiga kelas berikut: (1) kelas semua himpunan bagian dari Y dengan dua elemen, (2) kelas semua himpunan bagian dari Z dengan dua elemen, dan (3) kelas dari semua himpunan bagian X dengan dua elemen sedemikian rupa sehingga setiap himpunan bagian di kelas ini memiliki satu elemen dari Y dan satu elemen dari Z. Kelas (1) dan (2) masing-masing memiliki himpunan C(n, 2). Sebuah elemen dari Y dapat dipilih dalam n cara dan sebuah elemen dari Z dapat dipilih dalam n cara. Jadi kelas (3) memiliki (n)(n) himpunan. Jadi banyaknya himpunan bagian dari X dengan dua elemen adalah C(n, 2) + C(n, 2) + (n)(n).

Masalah Alokasi dan Kombinasi Umum

Sekarang perhatikan kumpulan n objek (tidak harus berbeda) yang termasuk dalam k kelompok berbeda, seperti dalam hipotesis Teorema 1.2.1. n_1 objek identik dari grup 1 dapat ditempatkan di himpunan n lokasi (sehingga tidak ada lokasi yang menerima lebih dari satu objek) dengan cara C(n, n_1). Kemudian n_2 objek dari grup berikutnya dapat ditempatkan dengan cara C(n – n_2). Kami melanjutkan dengan cara ini sampai semua tempat terisi. Dengan aturan perkalian, banyak cara untuk mengisi semua n titik adalah:

$$C(n, n_1) \cdot C(n - n_1, n_2) \cdot C(n - n_1 - n_2, n_3) \\ \cdot \cdot \cdot \cdot \cdot C(n - n_1 - n_2 - \cdot \cdot \cdot - n_{k-1}, n_k)$$

dan bilangan ini dilambangkan dengan C(n; n_1 n_2, . . . , n_k).

Ada cara lain untuk melihat proses alokasi ini. Misalkan X adalah kumpulan dari n objek berbeda dan n objek tersebut dialokasikan ke k lokasi sehingga lokasi i mendapatkan ni objek (i = 1, 2, . . , k) dimana $n_1 + n_2 + \cdots + n_k = n$. Kemudian setiap n1 objek dapat dipilih dari X dan dialokasikan ke lokasi 1 dengan cara C(n, n1). Selanjutnya, dari sisa (n – n_1) objek di X, n_2 objek dapat dialokasikan ke lokasi 2 dengan cara C(n – n_1, n_2). Kami melanjutkan seperti ini sampai semua objek habis.

Kami memiliki hasil berikut yang menghubungkan permutasi dan kombinasi umum.

TEOREMA 2.3.3

$$P(n; n_1, n_2, . . . , n_k) = C(n; n_1, n_2, . . , n_k)$$

dimana $n_1 + n_2 + \cdots + n_k \leq n$.

Bukti:

$$C(n, n_1) = \frac{(n!)}{(n_1)!\,(n - n_1)!}$$
$$C(n - n_1, n_2) = \frac{(n - n_1)!}{(n_2)!\,(n - n_1 - n_2)!}$$
$$C(n - n_1 - n_2, n_3) = \frac{(n - n_1 - n_2)!}{(n_3)!\,(n - n_1 - n_2 - n_3)!}$$
$$\vdots$$

Mengalikan suku-suku di ruas kanan persamaan, kita mendapatkan C(n; n_1, n_2, n_3, . . , n_k) = (n!)/(n_1)!(n_2)! ⋯ (n_k)!, dan dengan demikian teorema ditetapkan. Amati bahwa jika semua objek di X berbeda dan jika kita mengambil r objek, kita memiliki P(n; 1, 1, . . , 1) = C(n; 1, 1, . . , 1), di mana 1 diulang r kali. Sekarang P(n; 1, 1, . . , 1) = P(n, r)/(1!)(1!) ⋯ (1!) = P(n, r) dan C(n; 1, 1, . . , 1) = C(n, 1) • C(n – 1, 1) ⋯ C(n – r + 1, 1) = n(n – 1) ⋯ (n – r + 1) dan kita sekali lagi melihat bahwa P(n, r) = n(n – 1)(n – 2) ⋯ (n – r + 1).

Akibat Wajar

$$C(n; r) = C(n; n – r) = C(n; r, n – r) = C(n, r) = C(n, n – r)$$

Bukti:

$$C(n; r) = C(n, r) \cdot C(n - r, n - r) = C(n, r)$$
$$C(n; n - r) = C(n, n - r) \cdot C(n - (n - r), r) = C(n, n - r) = C(n, r)$$
$$C(n; r, n - r) = C(n, r) \cdot C(n - r, n - r) = C(n, r)$$

[Perhatikan bahwa C(n, r) = C(n; r) = P(n; r), tetapi P(n, r) = (r!)P(n; r).]

Contoh 2.3.2

$$P(15;3,5,7) = \frac{(15!)}{(3!)(5!)(7!)} = \frac{P(15,8)}{(3!)(5!)} = P(15;3,5)$$

$$\begin{aligned} C(15;3,5,7) &= C(15,3)C(12,5)C(7,7) \\ &= C(15,3)C(12,5) = C(15;3,5) \end{aligned}$$

Juga,

$$C(15,3) = \frac{(15!)}{(3!)(12!)} \quad \text{and} \quad C(12,5) = \frac{(12!)}{(5!)(7!)}$$

Dengan demikian:

$$C(15;3,5,7) = \frac{(15!)}{(3!)(5!)(7!)} = P(15;3,5,7)$$

Teorema Multinomial

TEOREMA 2.3.4 (Teorema Multinomial)

Dalam istilah khas dalam ekspansi ($x_1 + x_2 + \cdots + x_k$)n variabel xi (i = 1, 2, . . , k) muncul ni kali (di mana $n_1 + n_2 + \cdots + n_k = n$) dan koefisien dari suku ini adalah C(n; n_1, n_2, . . , n_k).

Bukti:

Bagian pertama dari pernyataan jelas karena ekspansi adalah produk dari n ekspresi di mana setiap ekspresi adalah jumlah dari k variabel. Istilah tipikal di sini tidak lain adalah permutasi umum dari n objek dalam kumpulan X yang terdiri dari k grup, dan oleh karena itu koefisien dari suku tipikal ini adalah jumlah permutasi umum tersebut.

Contoh 2.3.3

Koefisien $a^3b^2c^6d^4$ pada pemuaian $(a + b + c + d)^{15}$ adalah (15!)/(3!)(2!)(6!)(4!).

Contoh 2.3.4 (Teorema Binomial)

Teorema multinomial ketika k = 2 dikenal sebagai teorema binomial, yang dapat dinyatakan sebagai $(x + y)^n = C(n, n - r)x^{n-r}y^r$ di mana r bervariasi dari 0 hingga n [Sisi kanan persamaan ini disebut ekspansi binomial dari $(x + y)^n$. Koefisien C(n, r) yang muncul pada pemuaian binomial disebut koefisien binomial.]

Koefisien binomial dari $(x + y)^n$ dapat dihitung jika kita mengetahui koefisien binomial dari (x + y)n – 1 dengan menggunakan rumus Pascal: C(n, r) = C(n – 1, r) + C(n – 1, r – 1). Jadi koefisien binomial dapat disusun dalam bentuk segitiga yang dikenal sebagai segitiga Pascal:

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

1 6 15 20 15 6 1

....

Dalam representasi ini, (n + 1) koefisien binomial berturut-turut dari ekspansi binomial (x + y)n muncul di baris kemudian. Perhatikan bahwa elemen tipikal dalam sebuah baris (selain yang pertama dan terakhir) adalah jumlah dari dua suku tepat di atas elemen yang muncul di baris sebelumnya—dan itulah isi rumus Pascal. Pada setiap baris elemen pertama dan terakhir adalah 1, menunjukkan fakta bahwa jika A adalah himpunan dengan n elemen, maka hanya ada satu himpunan bagian dari A dengan n elemen dan hanya ada subset dari A yang tidak memiliki elemen.

Mempartisi Himpunan Berhingga

Diberikan suatu himpunan A dengan kardinalitas n, masalah kombinatorial yang menarik adalah menemukan banyaknya cara A dapat dipartisi menjadi k himpunan bagian sehingga himpunan bagian Ai (i = 1, 2, . . , k) memiliki elemen-elemen tepat ni,. Misalnya, jika A = {1, 2, 3, 4, 5, 6}, masalahnya adalah menemukan banyak cara untuk membagi A menjadi (1) dua himpunan bagian sehingga yang satu memiliki 2 elemen dan yang lain memiliki 4 elemen atau (2) dua himpunan bagian sehingga masing-masing memiliki 3 elemen atau (3) tiga himpunan bagian sehingga masing-masing memiliki 2 elemen, dan seterusnya. Masalah ini ekuivalen dengan masalah alokasi pengalokasian n objek berbeda ke k lokasi yang dibahas sebelumnya ketika kardinalitas setiap himpunan dalam partisi berbeda seperti pada (1).

Ada cara C(6; 2, 4) untuk mengalokasikan 6 elemen dari A ke dua lokasi sehingga lokasi 1 mendapat 2 elemen dan lokasi 2 mendapat 4 elemen. Banyaknya cara mempartisi A menjadi dua himpunan sehingga satu himpunan memiliki 2 elemen dan himpunan lainnya memiliki 4 juga C(6; 2, 4). Tetapi ketika himpunan bagian dalam sebuah partisi memiliki kardinalitas yang sama, kita harus memperhatikan situasi di mana pengulangan terjadi. Misalnya, jika P = {1, 2, 3} dan Q = {4, 5, 6}, maka partisi {P, Q} dan partisi {Q, P} adalah sama. Tetapi pengalokasian P ke lokasi 1 dan Q ke lokasi 2 tidak sama dengan pengalokasian Q ke lokasi 1 dan P ke lokasi 2. Banyaknya partisi A menjadi dua himpunan bagian yang kardinalitasnya sama adalah C(6; 3, 3)/2 . Secara umum, kita mendapatkan hasil berikut, yang merupakan perluasan dari teorema alokasi dan aturan perkalian.

TEOREMA 2.3.5

Banyaknya cara untuk membagi suatu himpunan kardinalitas n ke dalam suatu kelas yang terdiri dari himpunan bagian pi yang masing-masing memiliki kardinalitas ni (i = 1, 2, . . , k) di mana tidak ada dua bilangan ni yang sama adalah:

$$\frac{(n!)}{(p_1!)(n_1!)^{p_1}(p_2!)(n_2!)^{p_2}\cdots(p_k!)(n_k!)^{p_k}}$$

Contoh 2.3.5

(a) Banyaknya cara mengalokasikan 43 siswa ke dalam 7 asrama yang berbeda sehingga dua asrama pertama mendapat masing-masing 5 siswa, tiga berikutnya masing-masing mendapatkan 6 siswa, asrama keenam mendapat 7 siswa, dan asrama ketujuh mendapat 8 siswa adalah:

$$\frac{(43!)}{(5!)(5!)(6!)(6!)(6!)(7!)(8!)}$$

(b) Banyaknya cara membagi 43 siswa menjadi 7 kelompok sehingga terdapat 5 siswa dalam setiap 2 kelompok, 6 siswa dalam setiap 3 kelompok, 7 siswa dalam satu kelompok, dan 8 siswa dalam satu kelompok adalah

$$\frac{(43!)}{(2!)(5!)(5!)\cdot(3!)(6!)(6!)(6!)\cdot(7!)\cdot(8!)}$$

2.4 LEBIH LANJUT TENTANG PERMUTASI DAN KOMBINASI

Jika X adalah himpunan dengan n elemen, kita tahu bahwa permutasi r dari X adalah susunan elemen dari X yang tidak ada elemen yang berulang. Demikian pula, kombinasi-r adalah pemilihan elemen dari X di mana tidak ada elemen yang berulang. Dalam kedua kasus r tidak dapat melebihi n. Jika kita mengizinkan pengulangan, tidak ada batasan pada r. (Karena X adalah himpunan, n elemen di dalamnya berbeda.) Barisan-r dari X adalah susunan r elemen dari X di mana elemen-elemen tersebut dapat berulang, tetapi urutan kemunculan elemen-elemen ini penting. Misalnya, aabdac dan aadbac adalah dua barisan 6 yang berbeda dari himpunan X = {a, b, c, d}. Setiap r-permutasi jelas merupakan r-sequence. Di sisi lain, setiap r-sequence dengan elemen yang berbeda adalah r-permutasi. Penerapan sederhana dari aturan perkalian menunjukkan bahwa banyaknya barisan r dalam suatu himpunan dengan n anggota adalah n^r.

Setiap kumpulan dari r objek (tidak harus berbeda) yang dipilih dari himpunan X dari n elemen disebut koleksi-r dari X. Tidak seperti r-selection, urutan pemilihan elemen tidak penting dalam r-collection. Koleksi 4 [a, a, b, c] tidak berbeda dengan 4 koleksi [a, b, c, a]. Keduanya mewakili 4-koleksi yang sama. Setiap r-kombinasi adalah r-koleksi. Jika elemen-elemen dalam kumpulan-r berbeda, maka itu adalah kombinasi-r. Misalnya, jika X = {a, b, c, d} dan jika r = 3, himpunan semua 3-pilihan dari X akan mencakup setiap himpunan bagian dari X dengan 3 elemen dan pilihan seperti {a, a, a} , {a, b, b}, {d, a, d} dan seterusnya. Sebaliknya, jika r = 5, himpunan {a, b, b, b, d} adalah salah satu cara untuk memilih 5 elemen dari X, dan tidak ada himpunan bagian dari X yang dapat menjadi 5 himpunan.

Diberikan suatu himpunan kardinalitas n dan bilangan bulat positif sembarang r, dalam berapa cara seseorang dapat memilih r elemen (dengan pengulangan) dari X? Inilah jawabannya.

TEOREMA 2.4.1

Jika X adalah himpunan kardinalitas n, maka jumlah r-koleksi dari X adalah C(r + n – 1, n – 1), di mana r adalah sembarang bilangan bulat positif.

Bukti:

Misal X = {1, 2, 3, . . . , n}. Biarkan u menjadi r-koleksi dari X di mana 1 berulang x_1 kali, 2 berulang x_2 kali, . . . , dan N berulang x_n kali. Koleksi-r ini dapat direpresentasikan sebagai:

$$u = [\underbrace{1 \cdots 1} \quad \underbrace{2 \cdots 2} \quad \underbrace{3 \cdots 3} \qquad\qquad \underbrace{n \cdots n}]$$

dimana notasi i ⋯ i berarti simbol i berulang sebanyak xi kali. Demikian pula, biarkan v menjadi r-koleksi lain di mana 1 berulang y_1 kali, 2 kali y_2 kali, . . . , dan n mengulangi y_n kali. Kemudian:

$$v = [\underbrace{1 \cdots 1} \quad \underbrace{2 \cdots 2} \quad \underbrace{3 \cdots 3} \qquad\qquad \underbrace{n \cdots n}]$$

dimana notasi i ⋯ i berarti simbol i berulang y_2 kali. Perhatikan bahwa pada representasi u maupun pada representasi v terdapat celah antara 1 dan 2, celah antara 2 dan 3, . . . , jarak antara (n – 1) dan n. Pada setiap representasi terdapat (n – 1) gap. Yang membedakan satu r-koleksi dari yang lain adalah di mana titik-titik kosong ini berada dalam representasi yang khas.

Setiap representasi memiliki r simbol dan (n – 1) celah. Jadi setiap representasi dapat dianggap sebagai himpunan r + n – 1 lokasi berbeda. Semua n – 1 kosong adalah identik. Alokasi dari (n – 1) blanko ini ke (r + n – 1) lokasi mendefinisikan r-koleksi. Jadi jumlah r-koleksi yang berbeda sama dengan banyaknya cara untuk mengalokasikan (n – 1) objek yang identik ke (r + n – 1) lokasi yang berbeda sehingga setiap lokasi menerima paling banyak satu objek. Angka ini adalah C(r + n – 1, n – 1), seperti yang kita lihat di Bagian 1.3.

Contoh 2.4.1

Misal X = {a, b, c, d}. Jumlah total 5 pilihan dari X adalah C(5 + 4 – 1, 4 – 1) = 56. Teorema berikut adalah versi setara dari Teorema 1.4.1.

TEOREMA 2.4.2

(a) Jumlah solusi yang berbeda dalam bilangan bulat nonnegatif dari persamaan linier (dalam n variabel) $x_1 + x_2 + \cdots + x_n = r$ adalah C(r + n – 1, n – 1).

(b) Banyaknya penyelesaian berbeda dalam bilangan bulat tak negatif dari pertidaksamaan linier (dalam n variabel) $x_1 + x_2 + \cdots + x_n \leq r$ adalah C(r + n, n).

(c) Banyaknya suku pada ekspansi multinomial dari $(x_1 + x_2 + \cdots + x_n)^r$ adalah C(r + n – 1, n – 1).

Bukti:

(a) Setiap solusi xi = si (i = 1, 2, , n) dalam bilangan bulat tak negatif sesuai dengan kumpulan r elemen (dari himpunan X yang terdiri dari n variabel) di mana xi mengulang si kali, di mana si r, dan sebaliknya. Banyaknya himpunan tersebut adalah C(r + n – 1, n – 1) menurut Teorema 1.4.1.

(b) Misalkan y adalah variabel nonnegatif sehingga $x_1 + x_2 + \cdots + x_n + y = r$. (y disebut variabel slack.) Kita sekarang memiliki persamaan linier dalam (n + 1) variabel. Penyelesaian bilangan bulat tak negatif dari persamaan di (n + 1) variabel adalah solusi bilangan bulat tak negatif dari pertidaksamaan dalam n variabel, dan sebaliknya. Jadi bilangan yang dibutuhkan adalah C(r + n, n).

(c) Setiap suku dalam ekspansi dapat dianggap sebagai produk dari n variabel di mana jumlah eksponen variabel adalah r. Oleh karena itu, jumlah suku dalam pemuaian sama dengan jumlah kumpulan r elemen dari himpunan X yang terdiri dari n variabel yang memungkinkan pengulangan.

Contoh 2.4.2

Di asrama sarjana ada beberapa mahasiswa baru, mahasiswa tahun kedua, junior, dan senior.

(a) Dalam berapa cara sebuah tim yang terdiri dari 10 siswa dapat dipilih untuk mewakili asrama?

(b) Dalam berapa cara sebuah tim yang terdiri dari 10 orang dapat dipilih sedemikian rupa sehingga memiliki paling sedikit satu mahasiswa baru, paling sedikit satu mahasiswa tingkat dua, paling sedikit dua junior, dan paling sedikit dua senior?

Bukti:

(a) Jika p,q,r, dan s adalah banyaknya siswa dari setiap kelas dalam tim tersebut, maka banyaknya cara tim tersebut dapat dipilih adalah sama dengan banyaknya solusi dalam bilangan bulat tak negatif dari persamaan p+q +r+s=1. Jadi jawabannya adalah C(13,3) = 286.

(b) Dalam hal ini $p > 0$, $q > 0$, $r > 1$, dan $s > 1$. Tulis $p = p' + 1$, $q = q' + 1$, $r = r' + 2$, dan $s = s' + 2$. Jadi banyaknya cara akan sama dengan banyaknya penyelesaian bilangan bulat tak negatif dari persamaan $p' + q' + r' + s' + 6 = 10$ dan jawabannya adalah C(7, 3) = 35.

Masalah Alokasi di Pengaturan Umum

Kami sekarang mempertimbangkan masalah mengalokasikan r objek identik ke n lokasi berbeda sehingga setiap lokasi dapat menampung sebanyak mungkin objek yang diperlukan. Dalam berapa banyak cara kita dapat mencapai ini? Jika jumlah benda yang ditempatkan di lokasi i adalah xi (di mana i = 1, 2, . . , n), setiap solusi dari persamaan $x_1 + x_2 + \cdots + x_n = r$ dalam bilangan bulat tak negatif sesuai dengan cara mengalokasikan r objek ini ke n lokasi, dan sebaliknya. Jadi ada cara C(r + n – 1, n – 1) untuk menempatkan r objek identik di n lokasi berbeda. Kami menggabungkan pengamatan ini dengan Teorema 1.4.1 dan 1.4.2 untuk membuat pernyataan berikut:

TEOREMA 1.4.3

Diketahui:

L = Banyaknya cara memilih r elemen (dengan pengulangan) dari suatu himpunan yang memiliki n elemen

M = Banyaknya cara mengalokasikan r objek identik ke n lokasi berbeda

N = Jumlah solusi dalam bilangan bulat tak negatif dari persamaan $x_1 + x_2 + \cdots + x_n = r$

Kemudian:

$$L = M = N = C(r + n - 1, n - 1)$$

Kami sekarang merangkum empat kasus permutasi dan kombinasi (tanpa atau dengan pengulangan) dari r elemen dari satu set n elemen yang berbeda dan menafsirkan hasil ini sebagai dua model penghitungan sebagai berikut.

Model pemilihan

Banyaknya cara memilih r elemen dari himpunan n elemen adalah:

1. P(n, r) jika elemen yang dipilih berbeda dan urutan pemilihannya penting.
2. C(n, r) jika elemen yang dipilih berbeda dan urutan pemilihannya tidak penting.
3. nr jika elemen yang dipilih belum tentu berbeda dan urutannya penting.
4. C(r + n – 1, n – 1) jika elemen yang dipilih belum tentu berbeda dan ordonya tidak penting.

Model alokasi

Banyaknya cara mengalokasikan r objek ke n lokasi berbeda adalah:

1. P(n, r) jika objeknya berbeda dan tidak ada lokasi yang dapat mengambil lebih dari satu objek.
2. C(n, r) jika objeknya identik dan tidak ada lokasi yang dapat mengambil lebih dari satu objek.
3. nr jika objeknya berbeda dan tidak ada batasan jumlah objek di suatu lokasi.
4. C(r + n – 1, n – 1) jika objeknya identik dan tidak ada batasan jumlah objek di suatu lokasi.

Kesimpulan ini dapat diringkas dalam Tabel 2.1.

Kami menyimpulkan bagian ini dengan teorema berikut, yang merangkum berbagai kasus alokasi yang dipertimbangkan sejauh ini.

TEOREMA 2.4.4

(a) Jika r paling banyak n, kumpulan r objek yang berbeda dapat dialokasikan ke n lokasi, sehingga tidak ada lokasi yang dapat menerima lebih dari satu objek dengan cara P(n, r).

(b) Sekumpulan r objek berbeda dapat dialokasikan ke n lokasi dengan nr cara jika tidak ada batasan jumlah objek yang dapat diterima oleh suatu lokasi.

(c) Jika r paling banyak n, kumpulan r objek identik dapat dialokasikan ke n lokasi sehingga tidak ada lokasi yang dapat menerima lebih dari satu objek dengan cara C(n, r).

(d) Kumpulan r objek identik dapat dialokasikan ke n lokasi sehingga lokasi f mendapatkan setidaknya pi objek dalam cara C(r – p + n – 1, n – 1), di mana $p = p_1 + p_2 + \cdots +$ hal. (Teorema 1.4.3 adalah kasus khusus ketika setiap pi adalah 0.)

(e) Misalkan ada k jenis objek sehingga tipe i memiliki ni objek (i = 1, 2, . . , k). Objek yang termasuk dalam tipe yang sama adalah identik dan dua objek yang termasuk dalam dua tipe yang berbeda tidak identik. Kemudian objek $n_1 + n_2 + \cdots + n_k$ ini dapat dialokasikan ke n lokasi sehingga tidak ada lokasi yang dapat menerima lebih dari satu objek dengan cara $P(n; n_1 n_2, . . , n_k)$.

(f) Kumpulan $n_1 + n_2 + \cdots + n_k$ objek yang berbeda dapat dialokasikan ke k lokasi sehingga lokasi i menerima tepat ni objek (i = 1, 2, . . , k) di $C(n; n_1 n_2, . . , n_k)$ cara.

(g) $P(n; n_1, n_2, . . . , n_k) = C(n; n_1 n_2, . . , n_k) = P(n, r)/[(n_1!)(n_2!) . . . (n_k !)]$ dimana $n_1 + n_2 + \cdots + n_k = r$.

Tabel 2.1 Rangkuman kasus alokasi yang dipertimbangkan sejauh ini

	Selection model (X is a set with n elements. Select r elements from X.)	Allocation model (X is a set of n distinct locations. Allocate a collection of r objects to these locations.)
$P(n, r)$	Number of ways of arranging r elements (selecting where order is important) from X such that elements selected do not repeat	Number of ways of allocating r distinct objects to the n locations such that no location can receive more than one object
$C(n, r)$	Number of ways of choosing r elements (selecting where order is not important) from X such that elements selected do not repeat	Number of ways of allocating r identical objects to the n locations so that no location can receive more than one object
n^r	Number of ways of arranging r elements from X such that elements selected may repeat (no restriction on r and order is important)	Number of ways of allocating r distinct objects to the n locations if a location can receive more than one object
$C(p, q)$, where $p = r + n - 1$, $q = n - 1$	Number of ways of selecting r elements from X such that elements may repeat (no restriction on r and order is not important)	Number of ways of allocating r identical objects to the n locations if a location can receive more than one object

2.5 PRINSIP PIGEONHOLE

Ini adalah prinsip yang sangat jelas dan terlihat sangat sederhana, seolah-olah tidak memiliki arti penting. Namun, dalam praktiknya ini sangat penting dan kuat karena generalisasinya melibatkan beberapa hasil yang mendalam dan mendalam dalam teori kombinatorial dan teori bilangan. Kami menggunakan prinsip pigeonhole ketika kami mengatakan bahwa dalam setiap kelompok yang terdiri dari tiga orang setidaknya dua orang berjenis kelamin sama. Misalkan departemen ilmu komputer yang baru dibentuk di sebuah perguruan tinggi memiliki 10 anggota fakultas dan hanya 9 kantor untuk menampung mereka. Kemudian ide yang mendasari di balik pernyataan yang jelas bahwa setidaknya satu kantor akan memiliki lebih dari satu penghuni lagi-lagi prinsip pigeonhole. Jika ada 19 anggota fakultas, bukan 10, setidaknya satu kantor akan memiliki lebih dari dua penghuni. Demikian pula, jika ada setidaknya 367 siswa di asrama, sama jelas bahwa setidaknya dua dari mereka akan memiliki hari ulang tahun yang sama. Dilaporkan bahwa kulit kepala manusia memiliki paling banyak 99.999 helai rambut. Jadi di kota mana pun yang penduduknya melebihi 4 juta akan ada setidaknya 41 orang (kulit kepala botak tidak memiliki rambut) dengan jumlah rambut yang sama! Kita dapat mengutip beberapa contoh seperti ini.

Ide dasar yang mengatur semua contoh ini adalah fakta sederhana yang dikenal sebagai prinsip lubang merpati Dirichlet, yang dinyatakan secara formal sebagai berikut: Jika n + 1 atau lebih merpati menempati n lubang merpati, akan ada lebih dari satu merpati di setidaknya satu lubang merpati. Lebih umum, jika kn + 1 atau lebih merpati menempati n sarang, akan ada lebih dari k merpati di setidaknya satu merpati, di mana k adalah bilangan bulat positif.

Berikut adalah beberapa contoh untuk menggambarkan prinsip ini.

Contoh 2.5.1

Dalam turnamen round-robin (di mana setiap pemain bermain melawan setiap pemain lain tepat satu kali), anggaplah bahwa setiap pemain menang setidaknya sekali. Maka setidaknya ada dua pemain dengan jumlah kemenangan yang sama. Misalkan ada n pemain. Jumlah kemenangan untuk seorang pemain adalah 1 atau 2 atau 3 . . . atau (n – 1). Angka-angka (n – 1) ini sesuai dengan (n – 1) pigeonholes di mana n pemain akan ditempatkan. Jadi setidaknya dua dari mereka harus berada di lubang merpati yang sama dan mereka memiliki jumlah kemenangan yang sama.

Contoh 2.5.2

Ada 18 asrama di kampus. Dekan mahasiswa ingin melakukan survei di salah satu aula ini tentang penggunaan mikrokomputer, dan untuk melakukan ini dia harus membentuk komite yang terdiri dari 5 mahasiswa dari aula yang dipilih untuk survei. Sebuah iklan di koran kampus meminta sukarelawan dari 18 aula ini. Setidaknya berapa banyak tanggapan terhadap iklan yang cukup sebelum dekan dapat memilih aula dan membentuk kepanitiaan?

Larutan. Jawabannya adalah (4) (18) + 1 = 73 dengan prinsip pigeonhole.

Contoh 2.5.3

Sebuah kantong berisi tepat 5 kelereng merah, 8 biru, 10 putih, 12 hijau, dan 7 kuning. Tentukan jumlah bola paling sedikit yang akan dipilih yang akan menjamin bahwa akan ada (a) paling sedikit 4 kelereng berwarna sama, (b) paling sedikit 6 kelereng berwarna sama, (c) paling sedikit 7 kelereng berwarna berwarna sama, dan (d) paling sedikit 9 kelereng berwarna sama. (Di sini setiap warna mewakili lubang merpati. Jumlah lubang merpati adalah n = 5.)

Bukti:

(a) Jika paling sedikit 4 kelereng berwarna sama, maka terdapat satu lubang burung merpati yang dihuni lebih dari 3. Jadi dengan menerapkan prinsip lubang burung merpati umum dengan k = 3, jumlah kelereng yang akan dipilih paling sedikit adalah (3) • (5) + 1 = 16.

(b) n = 5 dan k = 5. Jadi jumlahnya adalah 26.

(c) n = 5 dan k = 6. Perhatikan bahwa ada batas atas jumlah kelereng merah. Hanya ada 5 kelereng merah. Jadi dalam hal ini bilangan yang dibutuhkan adalah [(6) • (5) + 1] – (6 – 5) = 30.

(d) Sekarang n = 5 dan k = 8 dengan batas atas 5 untuk merah dan 7 untuk kuning. Jadi bilangannya adalah [(8) • (5) + 1] – (8 – 5) – (8 – 7) – (8 – 8) = 37.

Jika m dan n bilangan bulat positif, maka lantai m/n adalah bilangan bulat terbesar yang kurang dari atau sama dengan m/n dan plafon m/n adalah bilangan bulat terkecil yang lebih besar dari atau sama dengan m/n. (Misalnya, lantai 38/9 adalah 4 dan langit-langitnya adalah 5.) Perpanjangan prinsip pigeonhole berikut ini mudah dibuat.

TEOREMA 2.5.1

(a) Jika m burung merpati dialokasikan untuk n lubang burung merpati, maka paling sedikit satu lubang memiliki lebih dari k burung merpati, di mana k adalah lantai dari $(m - 1)/n$.

(b) Jika $m = p_1 + p_2 + \cdots + p_n - n + 1$ merpati (setiap pi adalah bilangan bulat positif) diberikan kepada n sarang, maka lubang merpati pertama memiliki paling sedikit p_1 burung merpati, atau lubang merpati kedua memiliki setidaknya p_2 merpati, . . . , atau lubang merpati ke-n memiliki setidaknya pn merpati.

Bukti:

(a) Sekarang $(n) \bullet (k) (m - 1) < m$. Jika jumlah merpati tepat $n \bullet k$, dimungkinkan untuk mengalokasikan k merpati ke setiap lubang. Tetapi jumlah merpati adalah m, yang lebih besar dari $n \bullet k$. Jadi setidaknya ada satu lubang dengan lebih dari k penghuni.

(b) Di sini k = lantai dari $[(p_1 + p_2 + \cdots + pn)/n] - 1$. Jadi $(k + 1)$ sama dengan atau lebih besar dari setidaknya satu dari n bilangan bulat.

Contoh 2.5.4

Sebuah kantong berisi tepat 6 kelereng merah, 5 putih, dan 7 biru. Tentukan jumlah kelereng paling sedikit yang akan dipilih yang akan memastikan bahwa paling sedikit 3 kelereng merah atau paling sedikit 4 kelereng putih atau paling sedikit 5 kelereng biru yang terambil.

Bukti:

Metode Pertama (Menggunakan Teorema 2.5.1):

Di sini $n = 3$, $p_1 = 3$, $p_2 = 4$, dan $p_3 = 5$. Jadi $m = (3 + 4 + 5) - 3 + 1 = 10$.

Metode Kedua:

Misalkan banyaknya kelereng merah, putih, dan biru yang akan dipilih berturut-turut adalah x, y, dan z. Kita mensyaratkan bahwa x paling sedikit 3 atau y paling sedikit 4 atau z paling sedikit 5. Situasi ini tidak akan terjadi jika x paling banyak 2 dan y paling banyak 3 dan z paling banyak 4, yang menyiratkan bahwa $x + y + z$ paling banyak 9. Jadi kita harus memilih paling sedikit 10 kelereng.

Contoh 1.5.5

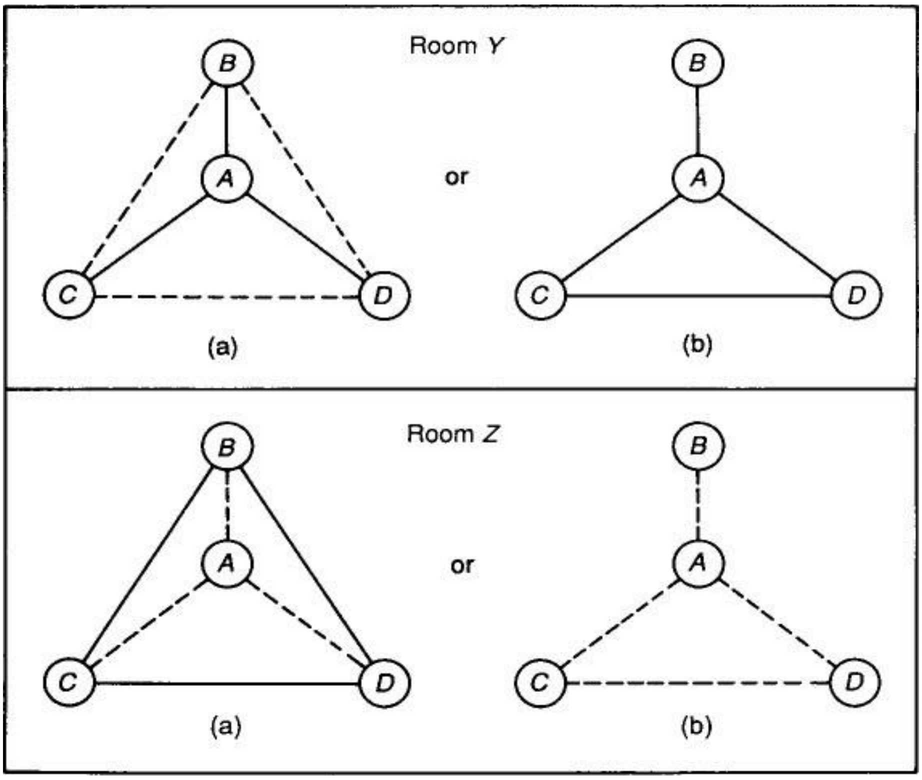
Dalam setiap kelompok yang terdiri dari 6 orang, ada 3 orang yang saling kenal atau ada 3 orang asing.

Bukti:

Misalkan {A, B, C, D, E, F} adalah himpunan yang terdiri dari 6 orang dan misalkan Y adalah sebuah ruangan di mana orang-orang yang diketahui A duduk. Misalkan Z adalah ruangan tempat duduk orang-orang yang tidak diketahui A. Lima individu B, C, D, E, dan F harus ditempatkan di dua ruangan Y dan Z. Jadi dengan proposisi sebelumnya baik Y atau Z memiliki setidaknya $k + 1$ individu, di mana k = lantai $(5 - l)/2 = 2$. Lihat Gambar 2.2. Jika ada garis putus-putus yang menghubungkan dua nama, kedua individu ini tidak saling mengenal. Jika ada garis yang menghubungkan dua individu, mereka saling mengenal.

(a) Misalkan ruangan Y memiliki 3 orang atau lebih. Misalkan B, C, dan D adalah tiga individu dalam Y. Ada dua kemungkinan: B, C, dan D tidak saling mengenal, seperti pada Gambar 2.2(a), membentuk kelompok yang terdiri dari 3 orang asing, atau pada setidaknya 2 dari mereka (misalnya, C dan D) saling mengenal, seperti pada Gambar 2.2(b). Dalam kasus terakhir, dua individu ini, C dan D, bersama dengan A, membentuk kelompok yang terdiri dari 3 orang yang saling mengenal.

(b) Misalkan ruangan Z memiliki 3 orang atau lebih. Misalkan B, C, dan D adalah 3 dari orang-orang di Z. Ada dua kemungkinan: Apakah 3 individu ini saling mengenal seperti pada Gambar 2.2(c), membentuk kelompok yang terdiri dari 3 individu yang saling kenal, atau ada setidaknya 2 individu (katakanlah, C dan D) yang tidak saling mengenal. Dalam kasus terakhir, 2 individu ini, C dan D, bersama dengan A, membentuk kelompok yang terdiri dari 3 orang asing.



Gambar 2.2 Ruang Y dan ruang Z

Kami menyimpulkan eksposisi singkat dari prinsip pigeonhole ini dengan dua teorema karena Paul Erdös.

TEOREMA 2.5.2

Misal X = {1, 2, 3, . . . , 2n) dan misalkan S adalah himpunan bagian dari X dengan (n + 1) elemen. Maka setidaknya ada dua angka dalam S sedemikian rupa sehingga yang satu membagi yang lain.

Bukti:

Setiap bilangan r dalam S dapat direpresentasikan sebagai $r = 2^t \bullet s$, di mana t adalah bilangan bulat tidak negatif dan s adalah bilangan ganjil dari X, disebut bagian ganjil dari r. Ada paling banyak n pilihan untuk s karena ada n bilangan ganjil di X. n bagian ganjil dapat dianggap sebagai n lubang merpati dan (n + 1) nomor S akan dialokasikan untuk lubang ini. Dengan kata lain, ada dua bilangan x dan y di S dengan bagian ganjil yang sama. Misalkan $x = 2^t \bullet s$ dan $y = 2^u \bullet s$. Kemudian x membagi y, atau sebaliknya.

TEOREMA 2.5.3

Setiap barisan dari (n2 + 1) bilangan yang berbeda mengandung barisan dari paling sedikit (n + 1) suku yang merupakan barisan naik atau barisan menurun.

Bukti:

Misalkan barisan tersebut adalah ai (i = 1, 2, . . . , n^2 + 1) dan misalkan ti adalah banyaknya suku pada barisan naik terpanjang yang dimulai dari ai. Jika ti, = n + 1 untuk beberapa i, kita selesai.

Misalkan ti n setiap i. Misalkan $H_j = \{a_i : t_i = j\}$, dimana j = 1, 2, . . . , n. Dengan demikian kita memiliki n pigeonholes $H_1, H_2, \ldots, H_n$ yang (n^2 + 1) nomor t_i, dialokasikan. Jadi dengan prinsip umum pigeonhole ada pigeonhole Hr yang berisi lebih dari k angka-angka ini di mana k = lantai [(n_2 + 1) – 1]/n = n. Jadi di antara angka-angka ti, setidaknya (n + 1) di antaranya sama.

Kami sekarang menetapkan bahwa (n + 1) angka dalam urutan yang sesuai dengan angka-angka ini di pigeonhole Hr membentuk urutan menurun. Biarkan ai dan aj berada di H_r, di mana i < j. Entah $a_i < a_j$ atau $a_i > a_j$ karena elemen-elemen dalam barisan semuanya berbeda. Misalkan ai < aj. Sekarang aj, ∈ Hr menyiratkan bahwa ada barisan panjang r mulai dari a_j. Jadi $a_i < a_j$ menyiratkan bahwa ada barisan panjang (r + 1) mulai dari a_j. Ini adalah kontradiksi, karena tidak mungkin ada barisan panjang (r + 1) dimulai dari a_i karena ai adalah elemen dari H_r. Jadi $a_i > a_j$ kapanpun i < j. Jadi setiap (n elemen dalam H_r akan menghasilkan turunan yang sangat menurun.

Contoh 2.5.6

Ilustrasikan Teorema 1.5.3 untuk barisan:

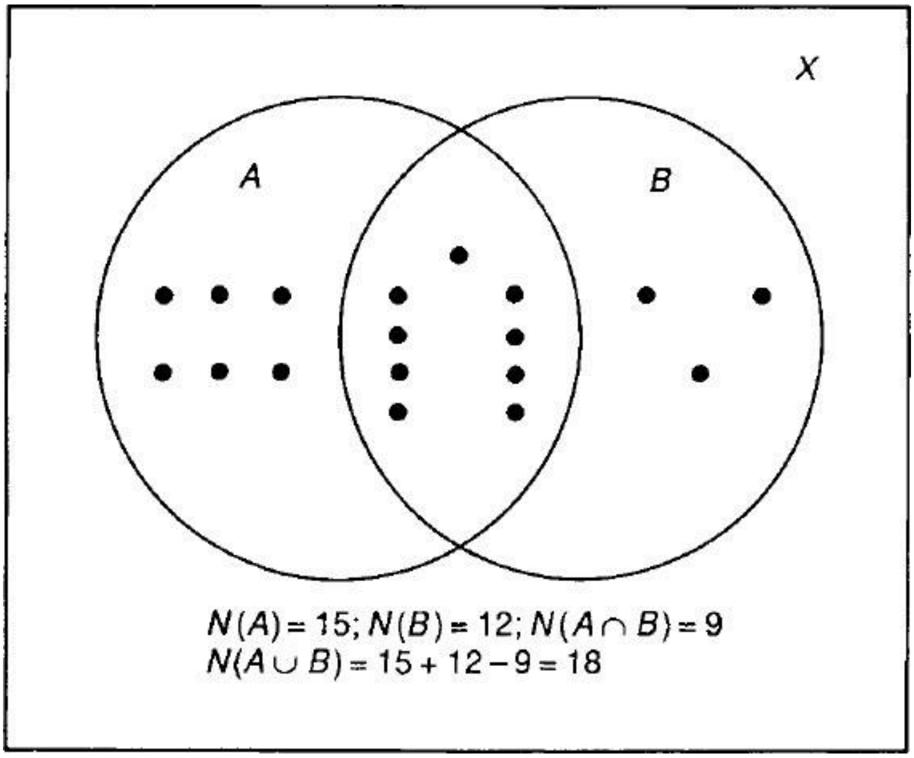
(a) 15, 12, 5, 7, 9, 6, 3, 4, 10, 14
(b) 15, 12, 9, 10, 7, 5, 4, 14, 3, 6

Bukti:

(a) Di sini n = 3, karena ada 10 elemen dalam barisan dan t_i yang bersesuaian (10 di antaranya) adalah 1, 2, 5, 4, 3, 3, 4, 3, 2, 1. Karena t_3 adalah 5 , terdapat kenaikan barisan 5 elemen dimulai dari a_3, yaitu 5, 7, 9, 10, 14.
(b) Di sini ti yang sesuai adalah 1, 2, 3, 2, 2, 3, 2, 1, 2, 1. Tidak ada yang melebihi 3. Kita mendapatkan H_1 = {15, 14, 6}, H_2 = {12 , 10, 7, 4, 3}, dan H_3 = {9, 5}. Barisan yang keluar dari himpunan kedua merupakan turunan menurun dari barisan yang diberikan dengan 5 angka.

2.6 PRINSIP INKLUSI

Jika X adalah sembarang himpunan berhingga, kita nyatakan dengan N(X) banyaknya anggota di X. Misalkan A dan B adalah dua himpunan berhingga yang tidak memiliki elemen yang sama. Maka, jelas, N(A B) = N(A) + N(B). Sebaliknya, jika perpotongan A dan B tidak kosong, untuk menghitung kardinalitas A ∪ B, pertama-tama kita cari jumlah N(A) + N(B) seperti sebelumnya. Dalam penjumlahan ini, unsur-unsur yang sama untuk A dan B dihitung (disertakan) dua kali—sekali saat menghitung N(A) dan kemudian saat menghitung N(B)—sehingga harus dihilangkan (dikecualikan) satu kali untuk mendapatkan jumlah total elemen dalam persatuan mereka.



Gambar 2.3 Inklusi

Sebagai contoh, jika dalam satu kelas terdapat 15 siswa yang mengambil mata kuliah kalkulus, 12 siswa yang mengambil matematika diskrit, dan 9 siswa yang mengambil kedua mata kuliah tersebut, maka banyaknya siswa yang mengambil minimal salah satu dari kedua mata kuliah tersebut adalah 15 + 12 – 9 = 18. Lihat diagram Venn pada Gambar 1.6.1 yang menyatakan himpunan semesta X dari semua siswa di kelas, himpunan A dari semua siswa di kelas yang mengambil kalkulus, dan himpunan B dari semua siswa yang mengambil matematika diskrit.

Fakta bahwa ada siswa di kelas yang mengambil kedua mata kuliah tersebut diperjelas dengan menunjukkan bahwa himpunan yang mewakili perpotongannya adalah daerah persekutuan untuk A dan B. Himpunan yang dikecualikan karena termasuk sekali dengan A dan kemudian dengan B adalah himpunan bagian $A \cap B$. Dengan demikian kita dapat menyatakan prinsip inklusi-eksklusi yang melibatkan dua himpunan berhingga sebagai berikut: Jika A dan B adalah dua himpunan berhingga, maka $N(A \cup B) = N(A) + N(B) - N(A \cap B)$.

Contoh 2.6.1

Dapatkan aturan inklusi-eksklusi yang melibatkan tiga himpunan berhingga. Misalkan A, B, dan C adalah tiga himpunan berhingga dan misalkan $D = B \cup C$. Sekarang $N(A \cup B \cup C) = N(A \cup D) = N(A) + N(D) - N(A \cap D)$ dan $N(D) = N(B \cup C) = N(B) + N(C) - N(B \cap C)$. Jadi:

$$N(A \cup B \cup C) = N(A) + N(B) + N(C) - N(B \cap C) - N(A \cap D) \quad (*)$$

Tapi $(A \cap D) = A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$. Jadi:

$$\begin{aligned} N(A \cap D) &= N(A \cap B) + N(A \cap C) - N((A \cap B) \cap (A \cap C)) \\ &= N(A \cap B) + N(A \cap C) - N(A \cap B \cap C) \end{aligned}$$

Dengan mensubstitusi N(A ∩ D) pada (*), kita mendapatkan

$$N(A \cup B \cup C) = [N(A) + N(B) + N(C)] - [N(A \cap B) + N(A \cap C) + N(B \cap C)] + N(A \cap B \cap C)$$

yang merupakan aturan inklusi-eksklusi untuk tiga set.

Misalkan himpunan yang kita pertimbangkan adalah semua himpunan bagian hingga dari himpunan hingga tertentu X dengan N elemen. Jika A adalah himpunan bagian dari X, komplemen dari A dilambangkan dengan A′. Maka A′ = X – A. Jadi:

$$N(A') = N - N(A) \qquad \text{for any subset } A \text{ contained in } X \qquad \text{(i)}$$

Selanjutnya misalkan A dan B adalah dua himpunan bagian dari X. Maka dengan (i), N((A ∪ B)′) = N – N(A ∪ B). Sekarang N(A ∪ B) = N(A) + N(B) – N(A ∪ B) dengan prinsip inklusi dan eksklusi. Dan (A ∪ B)′ = A′ ∩ B′. Dengan demikian:

$$N(A') = N - N(A) \qquad \text{for any subset } A \text{ contained in } X \qquad \text{(i)}$$

Demikian pula,

$$N(A' \cap B' \cap C') = N - N(A) - N(B) - N(C) + N(A \cap B) + N(A \cap C) + N(B \cap C) - N(A \cap B \cap C) \qquad \text{(iii)}$$

Kita dapat mempertimbangkan (i), (ii), dan (iii) juga sebagai aturan inklusi-eksklusi yang melibatkan satu himpunan bagian, dua himpunan bagian, dan tiga himpunan bagian dari himpunan dengan elemen N, masing-masing.

Sekarang anggaplah ai (i = 1, 2, 3) adalah tiga sifat berbeda yang diasosiasikan dengan elemen-elemen himpunan X sedemikian sehingga suatu elemen tipikal dapat memiliki satu atau lebih sifat-sifat ini atau mungkin tidak memilikinya. Misalkan Ai adalah himpunan semua x dalam X sedemikian sehingga x memiliki sifat ai. Misalkan $N(a_i)$ adalah jumlah elemen dalam X dengan properti ai N(ai′) adalah jumlah elemen dalam X yang tidak memiliki properti ai dan $N(a_1\ a_2)$ adalah jumlah elemen di X yang memiliki kedua properti a_1 dan a_2, dan seterusnya. Maka $N(a_i) = N(A_i)$, $N(a_i, a_j) = N(A_i \cap A_j)$, dan $N(a_i, a_j') = N(A_i \cap A_j')$. Aturan inklusi-eksklusi (iii) yang diberikan di atas yang melibatkan tiga himpunan bagian dari X dapat ditulis ulang sebagai berikut:

$$N(a_1', a_2', a_3') = N - [N(a_1) + N(a_2) + N(a_3)] + [N(a_1, a_2) + N(a_1, a_3) + N(a_2, a_3)] - N(a_1, a_2, a_3)$$

Hasil ini sekarang diperluas ke kasus yang melibatkan n sifat berbeda (yang mungkin dimiliki elemen-elemen dalam himpunan hingga) sebagai teorema yang dapat dibuktikan dengan menggunakan argumen kombinatorial. Sebelum memulai generalisasi ini, mari kita perkenalkan notasi berikut. Misalkan Ai (i = 1, 2, . . , n) adalah n himpunan bagian dari X. Perpotongan k-tupel adalah perpotongan dari k himpunan bagian yang berbeda dari n himpunan ini. Jumlah persimpangan k-tupel, tentu saja, C(n, k). Biarkan Sk menjadi jumlah dari jumlah elemen dari semua persimpangan k-tupel. Dengan demikian:

$$S_1 = N(A_1) + N(A_2) + \cdots + N(A_n)$$

$$S_2 = N(A_1 \cap A_2) + N(A_1 \cap A_3) + \cdots + N(A_{n-1} \cap A_n)$$

dan seterusnya.

TEOREMA 2.6.1 (Rumus Inklusi-Pengecualian)

$$N(A_1' \cap A_2' \cap A_3' \cap \cdots \cap A_n') = N - S_1 + S_2 - S_3 + \cdots + (-1)^n S_n$$

Bukti:

Untuk setiap elemen x di X dan untuk setiap subset A dari X, jumlah x di N(A) adalah 1 jika x ada di A. Jika tidak, jumlahnya adalah 0. Jadi cukup jika kita membuktikan bahwa jumlah elemen x dalam X adalah sama di kedua sisi persamaan.

(a) Misalkan x tidak berada pada salah satu dari n himpunan. Maka jumlah x pada ruas kiri (LHS) adalah tepat 1. Dan x ini memiliki bilangan 1 pada ruas kanan (Kanan) karena x adalah salah satu dari N elemen X, dan tidak dalam salah satu dari n set. Jadi, jumlah x yang tidak termasuk dalam salah satu dari n himpunan adalah 1 pada kedua sisinya.

(b) Misalkan x berada tepat di salah satu dari n himpunan. Maka hitungan x ini di ruas kiri adalah 0. Hitungan x di ruas kanan dihitung sebagai berikut: Jumlah x di N = 1, jumlah X di S_1 = 1, dan jumlah S_i = 0 ketika i tidak sama dengan 1. Jadi, jumlah x di ruas kanan adalah 1 – 1 = 0. Secara lebih umum, misalkan x adalah elemen yang sekutu bagi r dari n set. Hitungan x di ruas kiri, tentu saja, 0. Jumlah x di N = 1, jumlah x di S_1 = C(r, 1), jumlah x di S_2 = C(r, 2), dan seterusnya. Jadi jumlah x di ruas kanan

$$= 1 - C(r, 1) + C(r, 2) - \cdots + (-1)^{rC}(r, r) = (1 - 1)^r = 0$$

Jadi, jumlah x adalah sama pada kedua ruas persamaan, dan ini melengkapi pembuktian.

Contoh 2.6.2

Setiap mahasiswa di asrama mahasiswa baru mengambil setidaknya satu dari empat mata kuliah pengantar biologi (B), Bahasa Inggris (E), sejarah (H), dan matematika (M). Ada 6 mahasiswa yang mengambil keempat mata kuliah tersebut. Ada 25 siswa di masing-masing dari empat kursus, 15 siswa di dua dari empat kursus, dan 10 siswa di tiga dari empat kursus. Berapa banyak siswa yang ada di asrama?

Bukti:

Misalkan N adalah banyaknya siswa. Maka S_1 = C(4, 1) • (25) = 100, S_2 = C(4, 2) • (15) = 90, S_3 = C(4, 3) • (10) = 40, dan S_4 = C(4, 4) • (6) =6. Karena setiap siswa mengambil setidaknya satu mata kuliah, N(B′ E′ ∩ H′ M′) = 0. Jadi, dengan aturan inklusi-eksklusi, 0 = N – 100 + 90 – 40 + 6. Jadi N = 44.

Jika x adalah bilangan bulat positif apa pun yang kurang dari atau sama dengan bilangan bulat positif n, jumlah kelipatan x yang tidak melebihi n adalah lantai dari n/x, yang menurut definisi adalah bilangan bulat nonnegatif terkecil yang kurang dari atau sama dengan n/ x. Misalnya, jumlah bilangan bulat yang kurang dari 1000 dan habis dibagi 11 adalah 90, karena lantai 1000/11 = 90. Banyaknya bilangan bulat yang kurang dari 15 dan habis dibagi 11 adalah 1, yang merupakan lantai 15/11.

Contoh 2.6.3

Misal X = {1, 2, 3, . . . , 600}. Tentukan banyaknya bilangan bulat positif di X yang tidak habis dibagi 3 atau 5 atau 7.

Bukti:

Misalkan A, B, dan C adalah himpunan bilangan bulat di X yang masing-masing habis dibagi 3, 5, dan 7. Maka N(A) = lantai 600/3 = 200, N(B) = lantai 600/5 = 120, dan N(C) = lantai 600/7 = 85. Jadi S1 = 200 + 120 + 85 = 405.

Selanjutnya, N(A ∩ B) = jumlah bilangan bulat di X yang habis dibagi 15 = lantai 600/15 = 40. Demikian pula, N(A ∩ C) = lantai 600/21 = 28 dan N(B ∩ C) = lantai 600/35 = 17. Jadi S_2 = 40 + 28 + 17 = 85. Akhirnya, S_3 = N(A ∩ B ∩ C) = lantai 600/105 = 5. Jadi N(A′ ∩ B′ ∩ C) = 600 – 405 + 85 – 5 = 275. Jadi ada 275 bilangan pada himpunan yang tidak habis dibagi 3 atau 5 atau 7.

Dua bilangan bulat m dan n relatif prima jika satu-satunya pembagi positif yang memiliki kesamaan adalah 1. Kardinalitas himpunan bilangan bulat positif kurang dari n dan relatif prima ke n disebut fungsi total dari n dan dinotasikan dengan Φ (n). Misalnya, Φ (8) adalah kardinalitas dari himpunan {1, 3, 5, 7}, sehingga sama dengan 4.

Contoh 2.6.4

Gunakan aturan inklusi-eksklusi untuk menghitung Φ(60). Pembagi prima yang berbeda dari 60 adalah 2, 3, dan 5. Misalkan N(A), N(B), dan N(C) adalah banyaknya bilangan bulat yang kurang dari atau sama dengan 60 dan habis dibagi 2, 3, dan 5 , masing-masing. Maka N(A) = (60)/2, N(B) = (60)/3, dan N(C) = (60)/5. Juga, N(A ∩ B) = (60)/(2) (3), N(A ∩ C) = (60)/(2)(5), N(B ∩ C) = (60)/(3)(5), dan N(A ∩ B ∩ C) = (60)/(2)(3)(5). Dengan demikian:

$$\begin{aligned}\phi(60) &= 60 - \left[\frac{(60)}{2} + \frac{(60)}{3} + \frac{(60)}{5}\right] \\ &\qquad + \left[\frac{(60)}{(2)(3)} + \frac{(60)}{(2)(5)} + \frac{(60)}{(3)(5)}\right] - \frac{(60)}{(2)(3)(5)} \\ &= (60)\left(1-\frac{1}{2}\right)\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right) \\ &= \frac{(60)(2-1)(3-1)(5-1)}{30} = 16\end{aligned}$$

Contoh ini memiliki generalisasi langsung untuk sembarang bilangan bulat, sebagai berikut.

TEOREMA 2.6.2

Misalkan n bilangan bulat positif apa saja dan misalkan pi (i = 1, 2, . . , k) adalah faktor prima yang berbeda dari n. Maka (n) = (n/m) • 1) • (p_2 – 1) • . . . • (p_k – 1), di mana m adalah hasil kali dari k faktor prima yang berbeda dari n.

Contoh 2.6.5

Ada komputer mikro di masing-masing dari 32 kantor fakultas di departemen. Lima belas di antaranya memiliki monitor berwarna (C), 10 di antaranya memiliki printer laser (L), dan 8 di antaranya memiliki modem (M). Dua dari mereka memiliki ketiga opsi. Setidaknya berapa banyak yang tidak memiliki satu pun dari ketiga opsi tersebut?

Bukti:

Jika r adalah jumlah kantor tanpa opsi ini, r = 32 – [15 + 10 + 8] + [N(C ∩ L) + N(C ∩M) + N(L ∩ M)] – 2, di mana setiap bilangan yang tidak diketahui paling sedikit 2. Jadi r 32 – 33 + 6 – 2 = 3.

Contoh 2.6.6

Tentukan jumlah r solusi dalam bilangan bulat dari persamaan a + b + c = 25, di mana a paling sedikit 2 dan paling banyak 4, b paling sedikit 3 dan paling banyak 6, dan c paling sedikit 4 dan paling banyak 8 .

Bukti:

Angka yang diperlukan r adalah jumlah solusi dalam bilangan bulat dari persamaan yang direvisi x + y + z = 16, di mana batas atas untuk x, y, dan z berturut-turut adalah 2, 3, dan 4. Diketahui:

N = jumlah solusi dari persamaan yang direvisi
X = himpunan solusi sehingga x paling sedikit 3
Y = himpunan solusi sedemikian sehingga y paling sedikit 4
Z = himpunan solusi sedemikian sehingga z paling sedikit 5

Maka:

N = C(16 + 2, 2), N(X) = C(16 – 3 + 2, 2), N(Y) = C(16 4 + 2, 2), N(Z) = C(16 5 5 + 2, 2), N(X ∩ Y) = C(16 – 3 – 4 + 2, 2), N(X Z) = C(16 – 3 – 5 + 2, 2), N (Y ∩ Z) = C(16 – 4 – + 2, 2), dan N(X ∩ Y ∩ Z) = C(16 – 3 – 4 – 5 + 2, 2).

Dengan demikian:

$$\begin{aligned} r &= C(18, 2) - [C(15, 2) + C(14, 2) + C(13, 2)] \\ &\qquad + [C(11, 2) + C(10, 2) + C(9, 2)] - C(6, 2) \\ &= 153 - (105 + 91 + 78) + (55 + 45 + 36) - 15 = 0 \end{aligned}$$

Kekacauan

Misalkan X adalah himpunan berhingga dengan n elemen dan setiap elemen dalam himpunan tersebut diberi bilangan bulat positif unik (label) antara 1 dan n. Elemen yang diberi label i adalah elemen ke-i dari himpunan. Jika dalam permutasi elemen-elemen ini, elemen ke-i muncul pada posisi ke-i, elemen tersebut berada pada posisi semula untuk X. Derangement X adalah permutasi di mana tidak ada elemen yang muncul pada posisi semula. Sebagai contoh, misalkan X = {a, b, c, d} dan label dari a, b, c, d berturut-turut adalah 1, 2, 3, 4. Maka permutasi abdc bukanlah derangement, karena a dan b berada pada posisi semula. Tapi permutasi bade adalah kekacauan.

Sebagai contoh praktis, pertimbangkan skenario berikut: Seorang siswa mengirimkan lamaran pekerjaan ke berbagai agen perekrutan. Dia menyelesaikan 10 aplikasi berbeda yang ditujukan ke 10 agensi berbeda dan menulis alamat agensi ini pada 10 amplop identik. Dia kemudian menyuruh saudara laki-lakinya untuk memasukkan setiap aplikasi ke dalam amplop yang benar dan mengirimkan semua aplikasi ke agensi masing-masing. Dia memiliki kekacauan di tangan jika tidak ada aplikasi di dalam amplop yang tepat!

Jumlah total kekacauan dari himpunan kardinalitas n dilambangkan dengan Dn.

TEOREMA 2.6.3

Banyaknya derangement dari himpunan kardinalitas n adalah:

$$D_n = (n!)\left[1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \cdots + \frac{(-1)^n}{n!}\right]$$

Bukti:

Misalkan N adalah jumlah total permutasi pada X dan misalkan Ai adalah himpunan permutasi di mana objek ke-i berada di tempat asalnya. Banyaknya permutasi pada X adalah n!. Jadi D_n = n! – S_1, + S_2 + ⋯ + $(-1)^n S_n$,

Dimana:

$$\begin{aligned} S_1 &= C(n, 1) \cdot (n - 1)! \\ S_2 &= C(n, 2) \cdot (n - 2)! \\ &\vdots \\ S_k &= C(n, k) \cdot (n - k)! \end{aligned}$$

dan seterusnya. Jadi:

$$D_n = (n!)\left[1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \cdots + \frac{(-1)^n}{n!}\right]$$

Dengan menggunakan prinsip inklusi dan eksklusi, kita dapat memperoleh beberapa hasil penting yang melibatkan jumlah barisan r dan jumlah partisi dari suatu himpunan berhingga. Dua teorema berikut merupakan akibat langsung dari Teorema 1.6.1. Kita membutuhkan notasi sebelum teorema ini disajikan.

Jika n dan r adalah dua bilangan bulat positif ($n \leq r$), bilangan Stirling jenis kedua, dilambangkan dengan S(r, n), didefinisikan oleh hubungan berikut:

$$(n!) \cdot S(r, n) = n^r - C(n, 1) \cdot (n - 1)^r + C(n, 2) \cdot (n - 2)^r - \cdots + (-1)^{n-1} C(n, n - 1) \cdot 1^r$$

TEOREMA 2.6.4

Banyaknya barisan r yang dapat dibentuk dengan menggunakan elemen-elemen dari suatu himpunan dengan n elemen sedemikian rupa sehingga pada setiap barisan tersebut setiap elemen dari himpunan tersebut paling sedikit muncul satu kali adalah (n!) • S(r, n).

Bukti:

Misalkan X = {xi : i = 1, 2 , n} dan Ai adalah himpunan barisan-r yang tidak mengandung xi. Maka S_i, = C(n, i) • $(n\text{-}t)^r$

Hasil yang dinyatakan adalah konsekuensi langsung dari Teorema 2.6.1.

TEOREMA 2.6.5

(a) Masalah Alokasi. Banyaknya cara untuk mengalokasikan r objek berbeda ke n lokasi sehingga setiap lokasi menerima setidaknya satu objek adalah (n!) • S(r, n).

(b) Masalah Set Partisi. (1) Banyaknya partisi dari suatu himpunan kardinalitas r sehingga setiap partisi memiliki n himpunan tak kosong adalah S(r, n) dan (2) banyaknya partisi dari suatu himpunan dengan r elemen sehingga setiap partisi memiliki paling banyak n himpunan tak kosong adalah S(r, n) + S(r, n – 1) + + S(r, 1).

Bukti:

Kedua hasil ini mengikuti Teorema 2.6.1 dan definisi S(r, n).

Perhatikan perbedaan antara pernyataan dalam Teorema 2.3.5 dan Teorema 2.6.5(b).

Jumlah Fungsi dari Himpunan Hingga ke Himpunan Berhingga lainnya.

Misalkan X dan Y adalah dua himpunan berhingga dengan kardinalitas r dan n, berturut-turut. Misalkan f adalah sembarang fungsi arbitrer dari X ke Y. Salah satu dari r elemen dari X dapat dipetakan ke salah satu dari n elemen Y dengan n cara. Jadi dengan aturan perkalian ada nr fungsi dari X ke Y. Tetapi jika f adalah injeksi, jumlah cara elemen r dapat dipetakan jauh lebih sedikit: Faktanya, sama dengan n(n – 1)(n – 2) (n – r + 1). Jika pemetaan adalah surjeksi, setiap elemen di Y memiliki pragambar. Jadi oleh menerapkan Teorema 1.6.5 (a), kita melihat bahwa jumlah surjeksi adalah (n!) • S(r, n). Kita dapat meringkas hasil ini sebagai teorema.

TEOREMA 2.6.6

Misalkan X himpunan kardinalitas r dan Y himpunan kardinalitas n. Kemudian ada (1) nr fungsi dari X ke Y, (2) P(n, r) injeksi dari X ke Y, dan (3) (n!) • S(r, n) surjeksi dari X ke Y.

Generalisasi Prinsip Inklusi–Pengecualian

Bagian ini dapat dihilangkan tanpa kehilangan kontinuitas. Kami menyimpulkan bagian ini dengan teorema yang merupakan generalisasi dari Teorema 2.6.1. Untuk ini kita memerlukan beberapa notasi. Dengan setiap elemen x dari himpunan X dengan N elemen, kita mengasosiasikan bilangan real nonnegatif w(x) yang disebut bobot elemen. Kami memiliki satu set P dari n properti. Sebuah elemen khas di X mungkin atau mungkin tidak memiliki beberapa atau semua properti yang tercantum dalam P. V(r) adalah jumlah bobot semua elemen x di mana x memenuhi tepat r dari n properti ini. U(r) adalah jumlah bobot semua elemen x di mana x memenuhi setidaknya r dari n properti ini. Misal a,b,c. d, e elemen-elemen himpunan X dengan bobot masing-masing 6, 7, 8, 9, 10. P adalah himpunan empat sifat: s, t, u, v. Diketahui a memenuhi s dan t; b memenuhi s, u, dan v; c memenuhi t dan u; d memenuhi s, t, dan u; dan e memenuhi keempat properti. Maka V(1) = 0, V(2) = 6 + 8 = 14, V(3) = 7 + 9 = 16, dan V(4) = 10; dan U(1) = 6 + 7 + 8 + 9 + 10 = 40, U(2) = 40, U(3) = 26 dan U(4) = 10.

Akhirnya, jika Q adalah subset dari P, kita mendefinisikan bobot W(Q) dari himpunan Q sebagai jumlah bobot semua elemen x di mana x memenuhi semua r properti yang tercantum dalam Q, dan W(r) adalah jumlah semua suku bertipe W(Q), di mana Q adalah himpunan bagian dari P dengan r elemen. Misalnya, jika Q = {s, t}, maka W(Q) = w(a) + w(d) + w(e) = 25. W(2) adalah jumlah bobot kedua- himpunan bagian elemen dari P. Dalam contoh ini P memiliki enam himpunan bagian kardinalitas 2. Jadi W(2) = 25 + 26 + 17 + 27 + 10 + 17 = 122. W(0) menurut definisi adalah jumlah bobot semua elemen N di X.

TEOREMA 2.6.7 (Formula Inklusi-Pengecualian Umum)

(a) $V(r) = W(r) - C(r+1, r)W(r+1) + C(r+2, r)W(r+2) - \cdots + (-1)^{n-r}C(n, r)W(n)$. (Here $r \geq 1$.) (*)

(b) $U(r) = W(r) - C(r, r-1)W(r+1) + C(r+1, r-1)W(r+2) - \cdots + (-1)^{n-r}C(n-1, r-1)W(n)$. (Here $r > 1$.)

Bukti:

(a) Jelas, elemen tipikal x menyumbangkan bobotnya w(x) ke ruas kiri (*) jika dan hanya jika x memenuhi tepat r dari n sifat. Jadi cukup jika kita buktikan bahwa x menyumbang w(x) ke ruas kanan jika dan hanya jika x memenuhi tepat r dari n sifat.

Misalkan x memenuhi s dari n properti. Jika s = r, elemen x menyumbang w(x) ke W(r) di ruas kanan dan 0 ke suku lainnya. Jika s < r, kontribusi x ke ruas kanan adalah 0. Masih harus dibuktikan bahwa kontribusi x ke ruas kanan adalah 0 ketika s > r. Dalam hal ini, kontribusi x terhadap W(r) adalah C(s, r)w(x). Kontribusi x terhadap W(r + 1) adalah C(s, r + 1)w(x), dan seterusnya. Jadi ketika s > r, kontribusi x ke ruas kanan adalah:

$$w(x)[C(s, r) - C(s, r+1)C(r+1, r) + C(s, r+2)C(r+2, r) - \cdots + (-1)^{s-r}C(s, s)C(s, r)]$$

Sekarang mudah diverifikasi bahwa:

C(i, j)C(j, k) = C(i, k)C(i – k, i – j)

Jadi kontribusi x ke ruas kanan adalah w(x) • C(s, r) • K

Dimana:

$$\begin{aligned} K &= 1 - C(s-r, s-r-1) + C(s-r, s-r-2) - \cdots \\ &= 1 - C(s-r, 1) + C(s-r, 2) - \cdots \\ &= (1-1)^{s-r} = 0 \end{aligned}$$

Ini melengkapi bukti (a). Jika kita menempatkan r = 0 dan w(x) = 1 untuk setiap elemen x dalam X, kita mendapatkan rumus dalam Teorema 1.6.1 sebagai kasus khusus.

(b) Ini dibiarkan sebagai latihan.

Contoh 2.6.7

Verifikasi rumus yang diberikan dalam Teorema 1.6.7 untuk data berikut: Unsur-unsur himpunan X adalah a, b, c, d, dan e dengan bobot 6, 7, 8, 9, dan 10. P adalah himpunan empat sifat s, t, u, v. Diketahui a memenuhi p dan q; b memenuhi s, u, dan v; c memenuhi t dan u; d memenuhi s, t, dan u; dan akhirnya, e memenuhi keempat properti.

Larutan. Kami telah menghitung yang berikut: V(1) = 0, V(2) = 14, V(3) = 16, V(4) = 10, U(1) = 40, U(2) = 40, U (3) = 26, U(4) = 10, dan W(2) = 122. Selanjutnya kita temukan bahwa W(1) = 32 + 33 + 34 + 17 = 116, W(3) = 19 + 10 + 17 +10 = 56, dan W(4) = 10.

Rumus (i) ketika r = 1:

$$\begin{cases} \text{LHS} = V(1) = 0 \\ \text{RHS} = W(1) - C(2, 1)W(2) + C(3, 1)W(3) - C(4, 1)W(4) \\ \phantom{\text{RHS}} = 116 - 244 + 168 - 40 = 0 \end{cases}$$

Rumus (i) ketika r = 2:

$$\begin{cases} \text{LHS} = V(2) = 14 \\ \text{RHS} = 122 - (3)(56) + (6)(10) = 14 \end{cases}$$

Rumus (i) ketika r = 3:

$$\begin{cases} \text{LHS} = V(3) = 16 \\ \text{RHS} = 56 - (4)(10) = 16 \end{cases}$$

Rumus (i) ketika r = 4:

$$\begin{cases} \text{LHS} = V(4) = 10 \\ \text{RHS} = 10 \end{cases}$$

Rumus (ii) ketika r = 2:

$$\begin{cases} \text{LHS} = \text{U}(2) = 40 \\ \text{RHS} = 122 - (2)(56) + (3)(10) = 40 \end{cases}$$

Rumus (ii) ketika r = 3:

$$\begin{cases} \text{LHS} = U(3) = 26 \\ \text{RHS} = 56 - (3)(10) = 26 \end{cases}$$

Rumus (ii) ketika r = 4:

$$\begin{cases} \text{LHS} = U(4) = 10 \\ \text{RHS} = 10 \end{cases}$$

2.7 RINGKASAN HASIL DALAM KOMBINATORIKA

Kami sekarang menyimpulkan bab ini dengan daftar lengkap dari semua hasil penting yang melibatkan permutasi, kombinasi, alokasi, derangements, partisi himpunan, dan jumlah pemetaan antara set hingga yang dibuat di halaman ini:

1. Permutasi adalah pengaturan linier objek di mana urutan kemunculan objek yang berbeda sangat penting, sedangkan kombinasi hanyalah kumpulan objek dan urutan objek yang dipilih untuk dimasukkan di dalamnya tidak relevan.
2. (a) Banyaknya r-permutasi dengan r elemen berbeda yang dapat dibentuk menggunakan elemen dari kumpulan n elemen berbeda, (b) banyaknya cara mengalokasikan r objek berbeda ke n lokasi sehingga tidak ada lokasi yang dapat menerima lebih dari satu objek, dan (c) jumlah injeksi dari himpunan r elemen ke himpunan n elemen semuanya sama dengan $P(n, r) = n!/(n - r)!$
3. (a) Banyaknya r-kombinasi dengan r elemen berbeda yang dapat dibentuk dengan menggunakan elemen-elemen dari kumpulan n elemen berbeda dan (b) banyak cara untuk mengalokasikan r objek identik ke n lokasi sehingga tidak ada lokasi yang dapat menerima lebih dari satu objek keduanya sama dengan $C(n, r) = n!/[r!(n - r)!] = C(n, n - r)$.
4. Koefisien xr dalam $(1 + x)^n$ adalah $C(n, r)$.
5. (a) Banyaknya barisan r dengan r elemen (belum tentu berbeda) yang dapat dibentuk dengan menggunakan elemen dari kumpulan n elemen berbeda, (b) banyaknya cara menempatkan r objek berbeda di n lokasi dengan tidak ada batasan jumlah objek yang dapat diterima suatu lokasi, dan (c) jumlah fungsi dari himpunan r elemen ke himpunan n elemen semuanya sama dengan nr.
6. (a) Banyaknya r-koleksi dengan r elemen (belum tentu berbeda) yang dapat dibentuk dengan menggunakan elemen-elemen dari kumpulan n elemen berbeda, (b) banyaknya cara mengalokasikan r objek identik ke n lokasi dengan tidak ada batasan jumlah objek yang dapat diterima suatu lokasi, (c) jumlah solusi dalam bilangan bulat tak negatif dari $x_1 + x_2 + \cdots + x_n = r$, dan (d) jumlah suku pada perluasan $(x_1 + x_2 + \cdots + x_n)^r$ semuanya sama dengan $C(r + n - 1, n - 1)$.
7. Jika p_1 $p_2, \ldots,$ pn adalah bilangan bulat tak negatif yang jumlahnya p, maka (a) banyaknya r-kumpulan dengan r elemen (belum tentu berbeda) yang dapat dibentuk dengan menggunakan n elemen berbeda dari $X = \{x_1\ x_2, \ldots, x_n\}$ sedemikian rupa sehingga dalam setiap kombinasi xi muncul paling sedikit pi kali ($i = 1, 2, \ldots, n$), (b) banyaknya cara pengalokasian r objek identik ke n lokasi sehingga lokasi i paling sedikit benda pi ($i = 1, 2, \ldots, n$), dan (c) banyaknya solusi dalam bilangan bulat tak negatif dari $x_1 + x_2 + \cdots + x_n = r$ di mana xi paling sedikit p_1 ($i = 1, 2, \ldots, n$) semuanya sama dengan $C(r - p + n - 1, n - 1)$.
8. Tentukan $P(t; t_1, t_2, \ldots, t_j) = P(t, s)/[(t_1!)(t_2!) \cdots (t_j!)]$ di mana $s = t_1 + t_2 + \cdots + t_j$.
9. Jika terdapat n objek dari k tipe yang berbeda sehingga objek pada setiap tipe adalah identik dan jika tipe i memiliki ni objek ($i = 1, 2, \ldots, k$) maka ada $P(n; n_1, n_2, \ldots,$ nk) cara mengatur n objek ini dalam satu garis.

10. Jika ada k jenis objek dan jika tipe i memiliki ni objek yang identik (i = 1, 2, . . , k), maka ada P(n; n_1 n_2, . . , n_k) cara pengalokasian ini $n_1 + n_2 + \cdots + n_k$ objek ke n lokasi sehingga tidak ada lokasi yang dapat menerima lebih dari satu objek.
11. Ada cara P(n; n_1 n_2, . . , n_k) untuk mengalokasikan n objek berbeda ke k lokasi sehingga lokasi i tepat ni objek untuk setiap i = 1, 2, . . . , k.
12. Koefisien $x_1^{n1}\ x_2^{n2} \cdots x_k^{nk}$ di $(x_1 + x_2 + \cdots + x_k)^n$ adalah P(n; n_1, n_2, . . , n_k).
13. Suatu himpunan kardinalitas n dapat dipartisi menjadi suatu kelas yang terdiri dari himpunan bagian p_1 yang masing-masing memiliki kardinalitas n_1, himpunan bagian p_2 yang masing-masing memiliki kardinalitas n_2, . . . , dan p_k mensubset setiap kardinalitas nk dalam $\{n!)/\{[(p_1!)(n_1!)p_1][(p_2!)(n_2!)p_2] \bullet . . . \bullet [(p_k !)(n_k !)p_k]\}$ cara dimana bilangan bulat n_1 n_2, . . . , n_k berbeda.
14. Ketika n dan r adalah bilangan bulat positif, bilangan Stirling jenis kedua, dilambangkan dengan S(r, n), didefinisikan sebagai $(n!) \bullet S(r, n) = n^r – C\{n, 1) \bullet (n – 1)^r + C(n, 2) \bullet (n – 2)^r + \cdots + (–1)^{n–1}C(n, n – 1) \bullet 1^r$.
15. Banyaknya barisan r (elemen belum tentu berbeda) yang dapat dibentuk dengan menggunakan elemen himpunan X dari n elemen berbeda sehingga dalam setiap permutasi setiap elemen X muncul paling sedikit satu kali adalah (n!) • S(r, n).
16. Banyaknya cara untuk mengalokasikan r objek berbeda ke n lokasi sehingga setiap lokasi menerima setidaknya satu objek adalah (n!) • S(r, n).
17. Banyaknya partisi dari suatu himpunan dengan r elemen sedemikian rupa sehingga setiap partisi memiliki n himpunan tak kosong adalah S(r, n).
18. Banyaknya partisi dari suatu himpunan dengan r elemen sedemikian rupa sehingga setiap partisi memiliki paling banyak n himpunan tak kosong adalah $S(r, n) + S(r, n – 1) + \cdots + S(r, 1)$.
19. Banyaknya surjeksi dari himpunan r elemen ke himpunan n elemen adalah (n!) • S(r, n).
20. Banyaknya derangement dari suatu himpunan dengan n elemen adalah $D_n = (n!)[1 – 1/1! + 1/2! – 1/3! + \cdots + (– 1)n/n!]$.

Kombinatorik adalah salah satu cabang matematika yang paling dihormati. Rumus yang melibatkan pengaturan, barisan, dan kombinasi telah dikenal oleh matematikawan Cina, Hindu, dan Yunani sejak abad pertama Masehi. Kombinatorika terikat sangat erat dengan teori probabilitas, dan pada abad ketujuh belas dan kedelapan belas banyak matematikawan Eropa tertarik untuk mempelajarinya. dari probabilitas kombinatorial.

2.8 CATATAN DAN REFERENSI

Beberapa referensi umum yang sangat baik di bidang kombinatorik adalah buku-buku oleh Aigner (1979), Anderson (1979), Cohen (1978), Krishnamurthy (1986), Liu (1968), Riordan (1978), Roberts (1984), dan Tucker (1984). Ada juga teks klasik karya MacMahon (1960). Buku komprehensif pertama yang membahas permutasi dan kombinasi ditulis oleh Whitworth (1901). Bab 1 Grimaldi (1985), Bab 3 Liu (1985), dan Bab 2 Townsend (1987) juga membahas materi bab ini. Algoritma untuk menghasilkan permutasi dan kombinasi dari himpunan hingga yang diberikan diberikan secara rinci dalam Bab 1 dan 2 Genap (1973) dan dalam Bab 5 Reingold et al. (1977).

Seseorang tidak harus menjadi ahli matematika untuk mengetahui bahwa jika ada lebih banyak objek (merpati) daripada wadah (lubang merpati), akan ada setidaknya satu wadah dengan dua atau lebih benda. Perhatikan bahwa ini adalah pernyataan eksistensial: Ini hanya menegaskan bahwa ada wadah dengan setidaknya dua objek. Baik wadah maupun objeknya tidak dapat diidentifikasi. Tercatat bahwa Gustav Dirichlet (1805-1858) yang menggunakan prinsip ini secara ekstensif dalam penyelidikannya tentang masalah-masalah dalam teori bilangan—maka nama prinsip pigeonhole Dirichlet, yang juga dikenal sebagai 'prinsip kotak sepatu'. Generalisasi nontrivial dari prinsip yang tidak berbahaya ini melibatkan beberapa hasil yang paling mendalam dan mendalam dalam semua teori kombinatorial. Contoh 1.5.5 adalah kasus yang sangat khusus dari hasil yang dikenal sebagai teorema Ramsey. Referensi teori Ramsey termasuk Bab 5 dari Cohen (1978), Bab 8 dari Roberts (1984), Bab 4 dari Ryser (1963), dan buku tentang teori Ramsey oleh Graham et al. (1980).

Karya perintis menggunakan prinsip inklusi-eksklusi dilakukan oleh James Sylvester (1814–1897) dan pentingnya serta kegunaannya dipublikasikan dengan penerbitan buku Choice and Chance oleh Whitworth (1901). Untuk pembahasan prinsip ini, lihat Bab 5 dari Anderson (1979), Bab 5 dari Cohen (1978), Bab 7 dari Grimaldi (1985), Bab 4 dari Liu (1968), Bab 3 dari Riordan (1978), Bab 6 dari Roberts (1984), Bab 2 dari Ryser (1963), dan Bab 8 dari Tucker (1984).

2.9 LATIHAN SOAL

2.1 Nomor jaminan sosial seseorang adalah urutan sembilan digit yang belum tentu berbeda. Jika X adalah himpunan semua nomor jaminan sosial, tentukan jumlah elemen dalam X.

2.2 Ada enam karakter—tiga huruf alfabet Inggris diikuti tiga digit—yang muncul di panel belakang merek tertentu printer sebagai nomor identifikasi. Jika X adalah himpunan semua nomor identifikasi yang mungkin untuk merek printer ini, temukan jumlah elemen dalam X jika (a) karakter dapat diulang dalam nomor identifikasi, (b) digit tidak dapat diulang, (c) huruf tidak dapat diulang, dan (d) karakter tidak dapat diulang.

2.3 Tentukan banyak cara pengambilan (a) raja dan ratu, (b) raja atau ratu, (c) raja dan kartu merah, dan (d) raja atau kartu merah dari setumpuk kartu.

2.4 (a) Tentukan banyaknya bilangan genap antara 0 dan 100. (b) Tentukan banyaknya bilangan genap dengan angka berbeda antara 0 dan 100.

2.5 Urutan angka yang setiap angkanya 0 atau 1 disebut bilangan biner. Setiap digit dalam bilangan biner adalah komponen dari bilangan tersebut. Bilangan biner dengan delapan komponen disebut byte, (a) Temukan jumlah byte, (b) Temukan jumlah byte yang dimulai dengan 10 dan diakhiri dengan 01 . (c) Temukan jumlah byte yang dimulai dengan 10 tetapi tidak diakhiri dengan 01 . (d) Temukan jumlah byte yang dimulai dengan 10 atau diakhiri dengan 01.

2.6 Nama variabel dalam bahasa pemrograman BASIC adalah salah satu huruf alfabet atau huruf yang diikuti oleh angka. Temukan jumlah nama variabel yang berbeda dalam bahasa ini.

2.7 Urutan karakter disebut palindrom jika membaca dengan cara yang sama maju atau mundur. Misalnya, 59AA95 adalah palindrom enam karakter, dan 59A95 adalah palindrom lima karakter. Beberapa contoh palindrom lainnya: U NU, LON NOL, MALAYALAM, NOW ON, PUT UP, TOO HOT TO HOOT, NEVER ODD OR BAHKAN, MAMPU SAYA SEBELUM MELIHAT ELBA, dan MISKIN DAN DI DROOP. Temukan jumlah palindrom sembilan karakter yang dapat dibentuk dengan menggunakan huruf-huruf alfabet sedemikian rupa sehingga tidak ada huruf yang muncul lebih dari dua kali di masing-masingnya.

2.8 Tentukan banyak cara untuk membentuk barisan empat huruf dengan menggunakan huruf A, B, C, D, dan E jika (a) pengulangan huruf diperbolehkan, (b) pengulangan tidak diperbolehkan, (c) barisan mengandung huruf A tetapi pengulangan tidak diizinkan, dan (d) urutannya mengandung huruf A tetapi pengulangan diizinkan.

2.9 Dalam satu kelompok terdapat n pasangan suami istri. Tentukan banyak cara untuk memilih seorang wanita dan seorang pria yang bukan suaminya dari kelompok ini.

2.10 Misalkan X adalah himpunan semua polinomial berderajat 4 dalam satu variabel t sedemikian rupa sehingga setiap koefisien adalah bilangan bulat nonnegatif satu digit. Tentukan kardinalitas dari X.

2.11 Nama variabel dalam bahasa pemrograman FORTRAN adalah urutan yang memiliki paling banyak enam karakter sehingga karakter pertama adalah huruf alfabet dan karakter yang tersisa, jika ada, berupa huruf atau angka. Temukan jumlah nama variabel yang berbeda dalam bahasa ini.

2.12 Ada 10 anggota—A, B, C, D, E, F, G, H, I, dan J—dalam komite penggalangan dana. Tugas pertama panitia adalah memilih ketua, sekretaris, dan bendahara dari kelompok ini. Tidak ada individu yang dapat memegang lebih dari satu kantor. Tentukan banyak cara untuk memilih seorang ketua, sekretaris, dan bendahara sedemikian rupa sehingga (a) tidak ada yang keberatan untuk memegang salah satu dari ketiga jabatan tersebut, (b) C ingin menjadi ketua, (c) B akan tidak suka menjadi ketua, (d) A tidak suka menjadi ketua atau sekretaris, (e) I atau J ingin menjadi bendahara, dan (f) E atau F atau G ingin memegang salah satu dari ketiga kantor tersebut.

2.13 Ada tiga jembatan yang menghubungkan dua kota, A dan B. Antara kota B dan C ada empat jembatan. Seorang penjual harus melakukan perjalanan dari A ke C melalui B. Temukan (a) jumlah kemungkinan pilihan jembatan dari A ke C, (b) jumlah pilihan untuk perjalanan pulang-pergi dari A ke C, dan (c) jumlah pilihan untuk perjalanan pulang pergi jika tidak ada jembatan yang diulang.

2.14 Hitung (a) $P(8, 5)$, (b) $P(9, 2)$, dan (c) $P(6, 6)$.

2.15 Buktikan Teorema 1.2.2.

2.16 Tentukan nilai bilangan bulat positif n jika (a) $P(n, 2) = 30$, (b) $P(n, 3) = 24 \bullet P(n, 2)$, dan (c) $10 \bullet P(n, 2) = P(3n - 1, 2) + 40$.

2.17 Hitung 6!. Gunakan hasil ini untuk menghitung 7! dan 8!.

2.18 A dan B adalah dua anggota dalam sebuah pesta yang terdiri dari 12. Temukan banyak cara untuk menempatkan 12 orang ini ke 12 kamar yang terletak dalam satu baris sehingga setiap orang mendapat kamar dan (a) A dan B bersebelahan dan (b) A dan B tidak bersebelahan.

2.19 Tunjukkan bahwa $P(n, r + 1) = (n - r) \bullet P(n, r)$ dan gunakan hasil ini untuk mencari nilai n jika $P(n, 9) = 15 \bullet P(n, 8)$.

2.20 Tentukan nilai k jika $P(n + 1, r) = k \bullet P(n, r)$. Gunakan hasil ini untuk mencari n dan r jika $k = 5$, $n > r$, dan r sekecil mungkin.

2.21 Empat station wagon, lima sedan, dan enam van akan diparkir di 15 tempat parkir. Tentukan banyak cara untuk memarkir kendaraan tersebut sedemikian rupa sehingga (a) station wagon diparkir di awal, kemudian sedan, dan kemudian van, dan (b) kendaraan dari jenis yang sama diparkir secara en bloc.

2.22 Pertimbangkan koleksi enam batu dengan warna berbeda: biru (B), hijau (G), merah muda (P), merah (R), putih (W), dan kuning (Y). Tentukan (a) banyaknya cara membuat peniti untuk meletakkan batu-batu ini secara berjajar, (b) banyaknya cara membuat bros yang dengannya keenam batu tersebut akan dipasang membentuk pola melingkar, dan (c) banyaknya cara membuat cincin dengan menggunakan keenam batu tersebut.

2.23 Delapan orang akan duduk mengelilingi meja bundar besar untuk konferensi. Tentukan banyaknya kemungkinan susunan tempat duduk.

2.24 Seorang ibu dan dua anaknya yang masih kecil bergabung dengan tujuh anggota keluarganya untuk makan malam dan mereka harus duduk mengelilingi meja bundar. Tentukan banyaknya susunan tempat duduk yang memungkinkan agar kedua anak dapat duduk di kedua sisi ibu.

2.25 Enam anak perempuan dan enam anak laki-laki ditugaskan untuk berdiri di sekitar air mancur melingkar. Hitunglah jumlah tugas tersebut jika di kedua sisi anak laki-laki ada seorang gadis dan di kedua sisi anak perempuan ada anak laki-laki.

2.26 Jika X dan Y adalah dua himpunan dengan n anggota masing-masing dan jika tidak ada anggota yang sama pada kedua himpunan, tentukan banyak cara mengatur 2n elemen dari dua himpunan ini dalam pola melingkar sehingga di kedua sisi elemen X ada elemen Y, dan sebaliknya.

2.27 Hitung (a) $P(10; 4, 4, 2)$ dan (b) $P(12; 5, 4, 3)$.

2.28 Hitung (a) $P(17; 4, 3, 2)$ dan (b) $P(17; 2, 2, 2)$.

2.29 Buktikan bahwa jika m dan n bilangan bulat positif, $(mn)!(m!)n$ juga bilangan bulat positif.

2.30 Tentukan banyak cara agar kumpulan huruf lengkap yang membentuk kata MISSISSIPPI dapat disusun sedemikian rupa sehingga (a) tidak ada batasan letak huruf, dan (b) semua huruf S tetap bersatu.

2.31 Tentukan banyak cara untuk (a) menempatkan 9 siswa ke 11 kamar (diberi nomor urut dari 100 hingga 110) di asrama sehingga setiap kamar memiliki paling banyak satu penghuni, dan (b) memasang sembilan telepon berwarna (dua merah, tiga putih , dan empat biru) di kamar-kamar ini, sehingga setiap kamar memiliki paling banyak satu telepon.

2.32 Hitung (a) $C(9, 4)$, (b) $C(10, 7)$, dan (c) $C(8, 4)$

2.33 X adalah himpunan dengan sembilan elemen. Tentukan jumlah (a) himpunan bagian dari X, (b) himpunan bagian dari kardinalitas 3, dan (c) pasangan tak terurut pada X.

2.34 Buktikan rumus Pascal secara aljabar.

2.35 Ada 4 wanita dan 9 pria di fakultas matematika sebuah perguruan tinggi. Tentukan banyak cara untuk membentuk panitia perekrutan yang terdiri dari 2 wanita dan 3 pria dari departemen tersebut.

2.36 Ada 5 kemeja putih yang berbeda dan 7 kemeja biru yang berbeda di lemari pakaian. Tentukan banyak cara untuk mengambil 4 baju dari lemari sedemikian rupa sehingga (a) dapat berwarna putih atau biru, (b) semuanya putih, (c) semuanya berwarna biru, dan (d) semuanya berwarna warna yang sama, dan (e) 2 berwarna putih dan 2 berwarna biru.

2.37 Tentukan banyak cara tempat duduk r orang dari sekelompok n orang mengelilingi sebuah meja bundar.

2.38 Tentukan banyak cara tempat duduk 14 orang sehingga 8 orang mengelilingi satu meja bundar dan sisanya mengelilingi meja bundar lainnya.

2.39 Tentukan banyak cara tempat duduk 14 orang sehingga 8 orang berada di sekeliling meja bundar dan sisanya di atas bangku.

2.40 Temukan jumlah byte yang dapat dibentuk menggunakan tepat enam nol.

2.41 Tentukan banyaknya cara huruf-huruf yang muncul pada MISSISSIPPI dapat disusun kembali sehingga tidak ada dua huruf S yang bertetangga.

2.42 Dalam lotere negara bagian, sebuah tiket terdiri dari enam bilangan bulat berbeda yang dipilih dari himpunan X = {1, 2, 3, . . . , 42}. Pada setiap hari Sabtu pukul 20.00, enam bilangan bulat berbeda dipilih dari X oleh komputer. Pembeli tiket memenangkan (1) hadiah pertama (jackpot) jika enam angka dalam tiket sama dengan enam angka yang diambil oleh komputer, (2) hadiah kedua jika ada lima angka dalam tiket yang dipilih oleh komputer, (3) hadiah ketiga jika salah satu dari empat angka dalam tiket diambil oleh komputer, dan (4) hadiah keempat jika salah satu dari tiga angka dalam tiket diambil oleh komputer. Temukan (a) jumlah tiket berbeda yang harus dibeli seseorang yang pasti akan memastikan pembeli memenangkan jackpot dan peluang memenangkan jackpot jika seseorang membeli 1000 tiket, (b) peluang memenangkan hadiah kedua jika seseorang membeli satu tiket, dan (c) peluang memenangkan hadiah ketiga jika orang tersebut membeli satu tiket.

2.43 Buktikan identitas berikut menggunakan argumen kombinatorial:

$$\begin{aligned} C(n, r) = {} & C(r, r) \cdot C(n - r, 0) + C(r, r - 1) \cdot C(n - r, 1) \\ & + C(r, r - 2) \cdot C(n - r, 2) + C(r, r - 3) \cdot C(n - r, 3) + \cdots \\ & + C(r, 1) \cdot C(n - r, r - 1) + C(r, 0) \cdot C(n - r, r) \end{aligned}$$

2.44 Jika C(n, r) = C(r, 1) • C(n, r – 1), selesaikan n dalam bentuk r.

2.45 Buktikan bahwa C(pn, pn – n) adalah kelipatan dari p.

2.46 Buktikan identitas C(3n, 3) = 3C(n, 3) + 6n • C(n, 2) + n^3 menggunakan argumen kombinatorial.

2.47 Misalkan X adalah himpunan semua kata dengan panjang 10 yang huruf P muncul 2 kali, Q muncul 3 kali, dan R muncul 4 kali. Tentukan kardinalitas dari X.

2.48 Seorang ibu membeli 10 buku cerita untuk 3 anaknya. Yang termuda mendapat 2 buku dan dua lainnya masing-masing mendapatkan 4 buku. Temukan banyak cara dia dapat mengemasnya sebagai hadiah.

2.49 Sebuah kelas aljabar linier terdiri dari 10 jurusan matematika dan 12 jurusan ilmu komputer. Sebuah tim yang terdiri dari 12 orang harus dipilih dari kelas ini. Tentukan banyak cara untuk memilih sebuah tim jika (a) tim tersebut memiliki 6 dari setiap disiplin, dan (b) tim mayoritas memiliki jurusan ilmu komputer.

2.50 Temukan koefisien $a^2b^3c^3d^4$ dalam ekspansi (a) $(a + b + c + d)^{12}$ dan (b) $(2a – 3b + 2c – d)^{12}$.

2.51 Gunakan segitiga Pascal dan tuliskan koefisien dari suku-suku yang muncul dalam perluasan (x + y)n ketika n = 4, 5, dan 6.

2.52 Gunakan argumen kombinatorial untuk membuktikan identitas Newton: C(n, r) • C(r, k) = C(n, k) • C(n – k, r – k).

2.53 Buktikan identitas berikut:

$$C(n, 0) + C(n + 1, 1) + C(n + 2, 2) + \cdots + C(n + r, r) = C(n + r + 1, r)$$

2.54 Buktikan: C(n, 0) + C(n, 1) + C(n, 2) + ⋯+ C(n, n) = 2^n.

2.55 Gunakan argumen kombinatorial untuk membuktikan hal berikut:

[C(n, 0)]2 + [C(n, 1)]2 + [C(n, 2)]2 + ⋯+ [C(n, n)]2 = C(2n, n)

2.56 Buktikan identitas berikut:

$$C(m, 0), C(n, 0) + C(m, 1), C(n, 1) + C(m, 2) \cdot C(n, 2) + \cdots + C(m, n) \cdot C(n, n) = C(m + n, n)$$

2.57 Ada 18 siswa dalam satu kelas. Tentukan banyak cara untuk membagi kelas menjadi (a) 4 kelompok yang sama kuat dan kelompok minoritas, (b) 2 kelompok yang terdiri dari 5 siswa, 1 kelompok yang terdiri dari 4 siswa, dan 2 kelompok yang terdiri dari 2 siswa, dan (c) 1 kelompok 7 siswa, 1 kelompok 6 siswa, dan 1 kelompok 5 siswa.

2.58 Tentukan banyaknya r barisan yang dapat dibentuk dengan menggunakan anggota himpunan X = {A, B, C, D, E, F, G} jika (a) r = 4 dan unsur-unsur pada setiap barisan berbeda, (b) r = 4, dan (c) r = 9.

2.59 Tentukan banyaknya r-kumpulan yang dapat dibentuk dengan menggunakan anggota himpunan X = {A, B, C, D, E, F, G} jika (a) r = 4 dan elemen-elemen dalam setiap kumpulan berbeda, (b) r = 4, dan (c) r = 9.

2.60 Tentukan banyaknya penyelesaian berbeda dalam bilangan bulat tak negatif dari persamaan a + b + c + d + e = 24.

2.61 Temukan jumlah suku dalam ekspansi multinomial dari $(a + b + c + d + e)^{24}$.

2.62 Tentukan banyak cara untuk membentuk tim yang terdiri dari 15 mahasiswa dari sebuah universitas besar untuk mewakili mahasiswa baru, mahasiswa tahun kedua, junior, senior, dan mahasiswa pascasarjana sehingga tim tersebut memiliki (a) setidaknya satu dari setiap kelompok, (b) setidaknya dua dari masing-masing kelompok, dan (c) paling sedikit dua mahasiswa pascasarjana.

2.63 Tentukan banyaknya penyelesaian persamaan linear a + b + c + d + e= 10 jika (a) semua variabel adalah bilangan bulat tidak negatif, (b) semua variabel adalah bilangan bulat positif, dan (c) semua variabel positif bilangan bulat dan variabel a ganjil.

2.64 Tentukan banyak cara seorang ibu dapat membagikan 9 batang permen identik kepada ketiga anaknya sehingga setiap anak mendapat paling sedikit 2 batang.

2.65 Jumlah keempat bilangan bulat positif a, b, c, dan d paling banyak 10. Tentukan banyaknya pilihan yang mungkin untuk bilangan bulat tersebut.

2.66 Ketika sebuah dadu dilempar, salah satu dari enam bilangan bulat positif pertama diperoleh. Misalkan dadu dilempar lima kali dan jumlah dari lima bilangan bulat yang diperoleh ditambahkan. Lima lemparan merupakan percobaan. Tentukan banyaknya percobaan yang mungkin sehingga jumlahnya paling banyak 12.

2.67 Tentukan identitas berikut:

$$C(n, n) + C(n + 1, n) + C(n + 2, n) + \cdots + C(n + r, n) = C(n + r + 1, n + 1)$$

2.68 Tentukan banyaknya penyelesaian bilangan bulat tak negatif dari persamaan $x_1 + x_2 + x_3 + 3x_4 = 7$.

2.69 Jika X = {x_1, x_2, . . . , x_n} adalah kumpulan dari n objek berbeda dan r bilangan bulat positif, tentukan banyaknya r- koleksi X sedemikian rupa sehingga setiap koleksi tersebut memiliki objek xi diulang setidaknya pi kali di mana i = 1, 2, 3, . . . , n.

2.70 Tentukan banyak cara untuk mengalokasikan r objek identik ke n lokasi berbeda sehingga lokasi i mendapatkan paling sedikit pi objek, di mana i = 1, 2, . . . , n.

2.71 Tentukan banyaknya penyelesaian bilangan bulat tak negatif dari pertidaksamaan (ketat) a + b + c + d + e < 11.

2.72 Selesaikan Soal 1.71 jika a paling banyak 6.

2.73 Tunjukkan bahwa adalah mungkin untuk memiliki himpunan yang terdiri dari 5 orang sehingga tidak ada subgrup yang terdiri dari 3 orang asing atau subgrup yang terdiri dari 3 orang yang saling dikenal dalam himpunan ini.

2.74 Ada 4 penerbangan komuter dari kota A ke kota B setiap hari. Untuk hari tertentu, diketahui bahwa jumlah kursi kosong pada penerbangan ini masing-masing adalah 8, 10, 13, dan 9. Tentukan jumlah minimum tiket yang harus dijual sehingga jumlah kursi yang kosong adalah (a) paling banyak 1 di penerbangan 1 atau paling banyak 3 di penerbangan 2 atau paling banyak 6 di penerbangan 3 atau paling banyak 2 di penerbangan 4 , (b) paling banyak 2 di penerbangan 1 atau paling banyak 3 di penerbangan 2 atau paling banyak 4 di penerbangan 3 atau paling banyak 1 di penerbangan 4.

2.75 Buktikan bahwa dalam setiap kelompok yang terdiri dari 10 orang terdapat subgrup yang terdiri dari 3 orang asing atau subgrup yang terdiri dari 4 orang yang saling kenal.

2.76 Angka 1, 2, 3, . . . , n (n setidaknya 3) ditempatkan secara acak di sekitar lingkaran dan r adalah bilangan bulat yang kurang dari n. Membiarkan Si adalah jumlah r bilangan bulat berurutan (dianggap searah jarum jam) mulai dari i dan termasuk i, di mana i = 1, 2, . . ., n. Tunjukkan bahwa paling sedikit ada satu Si yang tidak lebih kecil dari lantai r(n + 1)/2.

2.77 Tunjukkan bahwa dalam setiap himpunan berhingga terdapat bilangan yang lebih besar dari atau sama dengan rata-rata aritmatika bilangan-bilangan dalam himpunan tersebut.

2.78 Misal X = {1, 2, 3, . . . , 600}. Tentukan banyaknya anggota X yang tidak habis dibagi 3 atau 5 atau 7.

2.79 (a) Dapatkan rumus untuk menemukan jumlah bilangan prima yang tidak melebihi bilangan bulat positif yang diberikan, (b) Gunakan rumus ini untuk menemukan jumlah bilangan prima yang tidak melebihi 100.

2.80 Bilangan bulat positif adalah bebas kuadrat jika tidak habis dibagi kuadrat dari bilangan bulat yang lebih besar dari 1. (a) Dapatkan rumus untuk menghitung jumlah bilangan bulat bebas kuadrat yang tidak melebihi bilangan bulat positif yang diberikan, dan (b) gunakan rumus ini untuk menghitung jumlah bilangan bebas kuadrat tidak melebihi 100.

2.81 Jika p dan q adalah dua bilangan prima yang berbeda, tentukan fungsi totien dari pq.

2.82 Ada enam kursi bertanda 1 sampai 6 di ruang konferensi kantor. Enam orang menghadiri seminar di ruangan ini pada pagi dan sore hari, (a) Tentukan jumlah permutasi dan derangements mengenai pengaturan tempat duduk, (b) temukan peluang bahwa tidak ada seorang pun yang duduk di kursi yang sama dua kali, (c) tentukan peluang bahwa tepat satu orang duduk di kursi yang sama dua kali, (d) temukan peluang bahwa setidaknya satu orang mendapat kursi yang sama dua kali, (e) temukan peluang bahwa tepat dua orang mempertahankan kursinya, dan (f) temukan probabilitas bahwa keenamnya mempertahankan kursi mereka.

2.83 Gunakan argumen kombinatorial untuk menetapkan identitas:

$$C(n, 0) \bullet Dn + C(n, 1) \bullet Dn{-}1, + C(n, 2) \bullet Dn{-}2 + \bullet \bullet \bullet + C(n, n) \bullet D0 = n!.$$

2.84 Tentukan banyaknya solusi bilangan bulat dari persamaan linear p + q + r = 25 di mana p paling sedikit 2 dan paling banyak 4, q paling sedikit 3 dan paling banyak 6, dan r paling sedikit 4 dan paling banyak 8.

2.85 Misalkan X adalah himpunan 4 barisan yang dapat dibentuk dengan menggunakan huruf A, B, C, D, E, dan F sehingga setiap barisan dalam X memiliki huruf A, B, dan C paling sedikit satu kali. Tentukan kardinalitas dari X.

2.86 Ada 5 lowongan pekerjaan di sebuah kantor. Berdasarkan tes tertulis dan wawancara pribadi, 4 kandidat dipilih dan masing-masing kandidat ditawari salah satu pekerjaan yang tersedia. Temukan banyak cara untuk menugaskan pekerjaan ini kepada para kandidat.

2.87 Tentukan banyaknya permutasi dari sembilan angka 1, 2, . . . , 9 di mana (a) balok 12, 34, dan 567 tidak muncul, dan (b) balok 12, 23, dan 415 tidak muncul.

2.88 Misalkan D(n, r) adalah banyaknya permutasi dari suatu himpunan yang terdiri dari n elemen di mana tepat r dari n elemen muncul pada posisi “alaminya” dan E(n, r) adalah jumlah permutasi di mana paling sedikit r dari n elemen muncul dalam posisi alaminya. Buktikan (a) D(n, 0) = Dn, (b) D(n, r) = C(n, r) • Dn–r, (c) D(n, n) = E(n, n) = 1, dan (d) jika S(i) = C(n, i) • (n – i)!, maka:

$$D(n, r) = S(r) - C(r+1, r) \cdot S(r+1) + C(r+2, r) \cdot S(r+2) - \cdots + (-1)^{n-r} C(n, r) \cdot S(n)$$

$$E(n, r) = S(r) - C(r, r-1) \cdot S(r+1) + C(r+1, r-1) \cdot S(r+2) - \cdots + (-1)^{n-r} C(n-1, r-1) \cdot S(n)$$

BAB 3
MENGHASILKAN FUNGSI

3.1 PENDAHULUAN

Dalam bab ini kami memperkenalkan konsep fungsi pembangkit—alat yang sangat berguna dalam memecahkan masalah penghitungan, khususnya masalah yang melibatkan pemilihan dan pengaturan objek dengan pengulangan dan dengan kendala tambahan. Pertimbangkan masalah persamaan bilangan bulat dari Bab 1, yang menanyakan jumlah solusi bilangan bulat nonnegatif dari $x_1 + x_2 + \cdots + x_n = r$, di mana kita tidak memberlakukan pembatasan lain pada variabel n. Bagaimana kita memecahkan masalah ini jika sekarang kita membatasi setiap variabel xi, menjadi elemen dari himpunan Vi? Soal umum: Temukan banyak cara untuk menghasilkan 62 sen yang melibatkan seperempat, dime, sen, dan sen. Penyelesaian adalah banyaknya penyelesaian pada bilangan bulat tak negatif dari q + d + n + c = 62, dimana q pada himpunan Q = {0, 25, 50}, d pada D = {0, 10, 20, 30 , 40, 50, 60}, n ada di N = {0, 5, 10, 15, 20, 25, + ⋯ + , 60}, dan c ada di C = {0, 1, 2, 3, 4, + ⋯ + , 60, 61, 62}. Sebelum kita mengembangkan prosedur untuk memecahkan "masalah pembuatan perubahan" ini menggunakan fungsi pembangkit, mari kita periksa masalah yang lebih sederhana.

Contoh 3.1.1

Tentukan banyaknya solusi bilangan bulat dari a + b + c = 10, dimana setiap variabel paling sedikit 2 dan paling banyak 4.

Solusi (Dengan Pencacahan Eksplisit):

a	*b*	*c*
2	4	4
3	4	3
3	3	4
4	2	4
4	4	2
4	3	3

Jadi ada enam solusi berbeda untuk masalah ini.

Sekarang kami memperkenalkan tiga polinomial p_a, p_b, dan p_c, satu untuk setiap variabel. Karena setiap variabel dapat berupa 2 atau 3 atau 4, dalam hal ini setiap polinomial didefinisikan sebagai $x^2 + x^3 + x^4$ dan kita mengalikan ketiga polinomial ini untuk memperoleh polinomial p(x) yang melibatkan pangkat x dengan eksponen berkisar antara 6 hingga 12. Polinomial p(x) ini adalah contoh fungsi pembangkit. Karena a + b + c = 10 kita sekarang mencari koefisien pangkat kesepuluh dari x dalam polinomial p(x). Dalam berapa cara kita dapat membentuk pangkat kesepuluh dari x dalam p(x)? Misalnya, kita dapat memilih x^2 dari pa, x^4 dari pb, dan x^4 dari pc dan mengalikannya. Ini hanyalah salah satu cara untuk mendapatkan pangkat kesepuluh dari x dan ini sesuai dengan solusi a = 2, b = 4, dan c = 4. Dengan kata lain, setiap solusi masalah sesuai dengan tepat satu cara untuk mendapatkan pangkat kesepuluh dari x dalam p(x). Jadi banyaknya penyelesaian soal adalah koefisien pangkat kesepuluh dari x dalam fungsi $p(x) = (x^2 + x^3 + x^4)^3$. Dengan perkalian polinomial biasa kita melihat bahwa koefisien ini adalah 6.

TEOREMA 3.1.1

(a) Deret pangkat adalah deret tak hingga dalam bentuk $a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots$, di mana ai (i = 0, 1, 2, . .) adalah bilangan real dan x adalah variabel.

(b) Jika $a_0 + a_1x + a_2x^2 + \cdots$ dan $b_0 + b_1x + b_2x^2 + \cdots$ adalah dua deret pangkat, maka (1) jumlah dua deret pangkat adalah deret pangkat dimana koefisien xr adalah ar + br dan (2) hasil kali dua deret pangkat adalah deret pangkat dengan koefisien xr adalah $(a_0b_r + a_1b_{r-1}, + a_2b_{r-2} + \cdots + a_rb_0)$.

(c) Jika r (r = 0, 1, 2, . . .) adalah banyaknya cara untuk memilih r objek dalam masalah kombinatorial tertentu (atau, lebih umumnya, jumlah solusi dari masalah kombinatorial), fungsi pembangkit biasa untuk masalah kombinatorial ini adalah deret pangkat $a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots +$.

Setiap polinomial di x adalah deret pangkat di x. Misalnya, polinomial $3x^2 + 2x^4$ dapat ditulis sebagai $0 + 0 \bullet x + 3x^2 + 0 \bullet x^3 + 2x^4 + 0 \bullet x^5 + 0 \bullet x^6 + \bullet \bullet \bullet \bullet +$. Prosedur penjumlahan dan perkalian dalam definisi adalah generalisasi yang jelas dari penjumlahan dan perkalian polinomial biasa.

Sekarang perhatikan masalah a + b + c = r, di mana a, b, dan c paling sedikit 2 dan paling banyak 4. Kemudian r bervariasi dari 6 hingga 12. Untuk pilihan r tetap, misalkan ar adalah banyaknya solusi dalam bilangan bulat. Maka ar adalah koefisien xr dalam fungsi pembangkit g(x) dari soal dimana $g(x) = (x^2 + x^3 + x^4)^3$, yang sama dengan $x^6 + 3x^7 + 6x^8 + 7x^9 + 6x^{10} + 3x^{11} + x^{12}$.

Contoh 3.1.2

Banyaknya cara memilih r elemen dari himpunan n elemen adalah C(n, r), sehingga fungsi pembangkit untuk masalah kombinatorial ini adalah g(x), di mana:

$$g(x) = C(n, 0) + C(n, 1)x + C(n, 2)x^2 + \cdots + C(n, r)x^r + \cdots C(n, n)x^n$$

yang merupakan ekspansi binomial untuk $(1 + x)^n$.

Contoh 3.1.3

Temukan fungsi pembangkit g(x) di mana koefisien x^r adalah ar, di mana ar adalah jumlah solusi dalam bilangan bulat tak negatif dari persamaan 2a + 3b + 5c = r.

Bukti:

Kami menulis A = 2a, B = 3b, dan C = 5c dan mencari jumlah solusi dari A + B + C = r, di mana A dalam himpunan {0, 2, 4, 6, . . .}, B ada di {0, 3, 6, 9, . . .}, dan C ada di {0, 5, 10, 15, . . .}. Jadi fungsi pembangkitnya adalah $g(x) = (1 + x^2 + x^4 + x^6 + . . .)(1 + x^3 + x^6 + x^9 + . . .)(1 + x^5 + x^{10} + x^{15} + . . .)$.

Contoh 3.1.4

Banyaknya solusi dalam bilangan bulat tak negatif dari a + b + c = 4 (tanpa kendala lain pada variabel) adalah koefisien dari x4 baik dalam $g(x) = (1 + x + x^2 + x^3 + x^4)^3$ atau dalam h(x)

$= (1 + x + x^2 + x^3 + x^4 + x^5 + \ldots)^3$. Perhatikan bahwa g(x) adalah polinomial di x, sedangkan h(x) adalah deret pangkat yang bukan polinomial.

Contoh 3.1.5

Jika ar adalah banyaknya cara memilih r kelereng dari kumpulan kelereng merah, biru, dan putih sehingga jumlah kelereng merah yang dipilih paling banyak dua, jumlah kelereng biru yang dipilih paling banyak tiga dan jumlah putih kelereng yang dipilih paling banyak empat, maka ar adalah koefisien xr pada fungsi pembangkit $g(x) = (1 + x + x^2)(1 + x + x^2 + x^3)(1 + x + x^2 + x^3 + x^4)$

Secara ekuivalen, koefisien xr dalam g(x) adalah banyaknya solusi dalam bilangan bulat tak negatif dari a + b + c = r, di mana a paling banyak 2, b paling banyak 3, dan c paling banyak 4.

3.2 FUNGSI PEMBANGKIT BIASA

Dalam Contoh 2.1.1, kita melihat bahwa jumlah langkah komputasi yang terlibat dalam menemukan jumlah solusi dengan pencacahan eksplisit sama persis dengan jumlah langkah komputasi yang terlibat dalam menemukan koefisien pangkat kesepuluh dari x dalam fungsi pembangkit, dan oleh karena itu metode fungsi pembangkit sama sekali tidak lebih efisien daripada metode pencacahan eksplisit. Sekarang kita akan mengembangkan beberapa teknik sederhana untuk menghitung koefisien fungsi pembangkit tanpa benar-benar melakukan prosedur perkalian polinomial (deret pangkat).

TEOREMA 3.2.1

(a) Misalkan ar adalah koefisien x^r dalam $g(x) = (1 + x + x^2 + x^3 + ..)^n$. Maka ar = C(r + n – 1, r).

(b) $(1 - x^m)^n = 1 - C(n, 1)x^m + C(n, 2)x^{2m} - + \cdots + (-1)^n x^{nm}$.

(c) $(1 + x + x^2 + \cdots + x^{m-1})^n = (1 - x^m)^n(1 + x + x^2 + ..)^n$.

Bukti:

(a) Fungsi g(x) adalah fungsi pembangkit yang terkait dengan masalah kombinatorial yang mencari jumlah ar solusi dalam bilangan bulat tak negatif dari persamaan $y_1 + y_2 + \cdots + y^n = r$ dan telah dibuktikan pada Bab 1 bahwa banyaknya solusi adalah C(r + n – 1, n – 1), yang sama dengan C(r + n – 1, r).

(b) Masukkan $t = (-x^m)$ ke dalam ekspansi binomial (1 + t)n.

(c) Mudah diverifikasi (dalam arti formal) bahwa

$$1 + x + x^2 + \cdots + x^{m-1} = (1 - x^m)(1 + x + x^2 + ...)$$

Sekarang ambil pangkat ke-n pada kedua ruas persamaan ini.

Contoh 3.2.1

Tentukan banyaknya penyelesaian bilangan bulat dari persamaan a + b + c + d = 27, dimana setiap variabel paling sedikit 3 dan paling banyak 8.

Larutan. Banyaknya solusi adalah koefisien pangkat dua puluh tujuh dari x dalam $g(x) = (x^3 + x^4 + \cdots + x^8)^4$, dan bilangan ini adalah koefisien pangkat lima belas x dalam $h(x) = (1 + x + \cdots + x^5)^4$. Dengan (c) dari Teorema 3.2.1:

$$h(x) = (1 - x^6)^4(1 + x + x^2 + \cdots +)^4$$

Dengan (b) dari teorema ini:

$$(1 - x^6)^4 = 1 - C(4, l)x^6 + C(4, 2)x^{12} + \cdots +$$

dan dengan (a) dari teorema yang sama:

$$(1 + x + x^2 + \cdots +)^4 = 1 + C(4, 1)x + C(5, 2)x^2 + C(6, 3)x^3 + \cdots +$$

Jadi koefisien pangkat lima belas x dalam h(x) sama dengan:

$$C(18, 15) - C(4, 1)C(12, 9) + C(4, 2)C(6, 3)$$

Contoh 3.2.2

Temukan koefisien pangkat dua puluh empat dari x dalam $(x^3 + x^4 + . . .)^5$.

Larutan. Angka yang diinginkan adalah koefisien pangkat kesembilan dari x dalam g(x) = $(1 + x + x^2 + . . .)^5$, yang sama dengan C(13, 4).

Jika $a_0 + a_1x + a_2x^2 + . . . +$ arxr $+ . . . +$ adalah perluasan deret pangkat dari suatu fungsi g(x), maka g(x) adalah fungsi pembangkit biasa untuk barisan a_r. Dari fungsi pembangkit yang diberikan dimungkinkan untuk membangun fungsi pembangkit baru untuk pilihan a_r yang berbeda, dan ini adalah isi dari teorema berikutnya, yang pembuktiannya ditinggalkan sebagai latihan.

TEOREMA 3.2.2

Jika g(x) adalah fungsi pembangkit untuk ar dan h(x) adalah fungsi pembangkit untuk br maka:

(a) Ag(x) + Bh(x) adalah fungsi pembangkit untuk $Aa_r + Bb_r$.
(b) (1 – x)g(x) adalah fungsi pembangkit untuk ar – a_{r-1}.
(c) $(1 + x + x^2 + . . .)g(x)$ adalah fungsi pembangkit untuk $(a_0 + a_1 + a_2 + \cdots + a_r)$.
(d) g(x)h(x) adalah fungsi pembangkit untuk $(a_0b_r + a_1b_{r-1} + a_2b_{r-2} + \cdots + a_rb_0)$.
(e) xg′(x) adalah fungsi pembangkit untuk rar, di mana g′(x) adalah turunan dari g(x) terhadap x.

Bila lambang x adalah bilangan real dengan nilai mutlak lebih kecil dari 1 maka dapat dibuktikan bahwa $(1 - x)(1 + x + x^2 + x^3 + . . .) = 1$. (Untuk pembuktian, lihat pembahasan konvergensi deret geometri dalam setiap buku pengantar kalkulus Dalam buku ini kita lebih tertarik pada koefisien pangkat x yang dianggap sebagai simbol daripada dengan masalah konvergensi.) Jadi kami menulis:

$$g(x) = 1 + x + x^2 + x^3 + \cdots = \frac{1}{1 - x} = (1 - x)^{-1}$$

$$h(x) = (g(x))^n = \left(\frac{1}{1 - x}\right)^n = (1 - x)^{-n}$$

di mana g(x) adalah fungsi pembangkit untuk ar = 1 dan h(x) adalah fungsi pembangkit untuk ar = C(r + n – 1, r).

Contoh 3.2.3

Tentukan fungsi pembangkit untuk ar = $3r + 5r^2$.

Bukti:

Misalkan g(x) = 1/(1 – x). Fungsi pembangkit untuk 1 adalah g(x). Jadi fungsi pembangkit untuk r adalah xg'(x), dengan (e) dari Teorema 2.2.2. Dengan menerapkan prinsip ini sekali lagi kita melihat bahwa fungsi pembangkit untuk r^2 adalah x(xg'(x))'. Jadi fungsi pembangkit yang diinginkan adalah 3xg'(x) + 5x(xg'(x))', yang sama dengan:

$$\frac{3x}{(1-x)^2} + \frac{5x + 5x^2}{(1-x)^3}$$

3.3 FUNGSI PEMBANGKIT EKSPONENSIAL

Fungsi pembangkit yang telah kita lihat sejauh ini disebut sebagai fungsi pembangkit "biasa" karena berhubungan dengan masalah pemilihan yang urutannya tidak relevan. Dengan kata lain, mereka digunakan untuk memecahkan masalah kombinatorial distribusi objek identik (tidak dapat dibedakan) ke lokasi yang berbeda. Sekarang kita beralih ke masalah pengaturan di mana urutan memainkan peran penting. Sebagai contoh, masalah mencari banyak cara agar 5 kelereng merah (tidak dapat dibedakan) dapat dimasukkan ke dalam 3 kotak berbeda adalah masalah yang urutannya tidak relevan, sedangkan masalah menemukan banyak cara menyusun 5 kelereng dalam deretan menggunakan tiga jenis kelereng yang berbeda (misalnya merah, biru, dan putih) adalah masalah di mana urutan memainkan peran penting. Susunan RRBBW (merah, merah, biru, biru, putih) tidak sama dengan susunan RBRBW, meskipun kedua susunan tersebut menggunakan jumlah kelereng merah, biru, dan putih yang sama. Fungsi pembangkit yang didefinisikan sehubungan dengan masalah kombinatorial seperti itu, di mana urutannya relevan, disebut fungsi pembangkit eksponensial. Mari kita menganalisis contoh mengatur kelereng sebelum kita memberikan definisi formal fungsi pembangkit eksponensial.

Contoh 3.3.1

Tentukan banyak cara menyusun 5 kelereng berjajar menggunakan kelereng tiga warna (merah, biru, dan putih) sehingga dalam setiap susunan paling sedikit ada satu kelereng setiap warna, dengan asumsi masing-masing paling sedikit ada 3 kelereng warna yang kami miliki.

Bukti:

Misalkan banyaknya kelereng merah, biru, dan putih pada susunan tertentu adalah r, b, dan w. Kemudian r + b + w = 5, di mana setiap variabel adalah bilangan bulat yang paling sedikit 1. Kita tahu (dari pembahasan kita tentang permutasi umum di Bab 1) bahwa dengan pilihan khusus ini atau r, b, dan w ada (5 !)/(r!)(b!)(w!) cara menyusun 5 kelereng berturut-turut. Jadi jumlah total susunan akan menjadi jumlah semua ekspresi dari bentuk ((r + b + w!))/(r!)(b!)(w!), di mana r + b + w = 5 dan masing-masing variabel adalah bilangan bulat yang paling sedikit 1. Pilihan r, b, dan w adalah sebagai berikut:

r	b	w
3	1	1
1	3	1
1	1	3
2	2	1
2	1	2
1	2	2

Jadi banyaknya susunan adalah:

$$\frac{(5!)}{(3!)(1!)(!!)} + \frac{(5!)}{(1!)(3!)(1!)} + \frac{(5!)}{(1!)(1!)(3!)} + \frac{(5!)}{(2!)(2!)(1!)} + \frac{(5!)}{(2!)(1!)(2!)} + \frac{(5!)}{(1!)(2!)(2!)} = 150$$

Sekarang dapat dengan mudah diverifikasi bahwa koefisien $x^5/5!$ dalam fungsi g(x), di mana $g(x) = (x/(l!) + x^2/(2!) + x^3/(3!))^3$ tepat jumlah dari enam ekspresi yang diperoleh di paragraf sebelumnya memberikan jumlah total pengaturan. Fungsi g(x) adalah contoh fungsi pembangkit eksponensial. Seperti dalam kasus fungsi pembangkit biasa, kita mengambil pangkat ketiga dari polinomial (mewakili tiga warna berbeda), dan pangkat variabel dalam polinomial adalah 1, 2, dan 3, yang menunjukkan bahwa banyak cara kelereng warna tertentu dapat muncul dalam suatu susunan adalah 1, 2, atau 3. Perbedaan signifikan di sini adalah bahwa tidak seperti fungsi pembangkit biasa, koefisien x^r dalam polinomial adalah l/(r!) dan solusi dari kombinasi masalahnya adalah koefisien $x^r/(r!)$ dalam fungsi pembangkit eksponensial. Apakah ada metode yang lebih mudah dalam soal ini untuk mencari koefisien $x^5/(5!)$ dalam g(x)? Misalkan $h(x) = (e^x - 1)^3$, di mana ex adalah fungsi eksponensial (deret pangkat) yang didefinisikan oleh $1 + x + x^2/(2!) + x3/(3!) + \cdots +$, di mana x adalah setiap variabel nyata. Maka koefisien yang dibutuhkan adalah $x^5/(5!)$ dalam $h(x) = (e^{3x} - 3e^{2x} + 3e^x - 1)$ dan koefisien ini adalah $3^5 - (3)2^5 + 3 = 150$.

TEOREMA 3.3.1

Jika br (r = 0, 1, 2, . .) adalah solusi dari masalah kombinatorial, deret pangkat g(x) didefinisikan oleh $b_0 + b_1x + (b_2x^2)/(2!) + (b_3x^3)/(3!) + \bullet\bullet\bullet +$ disebut fungsi pembangkit eksponensial untuk masalah tersebut.

Contoh 2.3.2

Temukan fungsi pembangkit eksponensial untuk br, banyaknya cara menyusun r elemen berbeda dari himpunan n elemen.

Bukti:

Tentu saja, br = P(n, r), jadi fungsi pembangkit eksponensial untuk soal ini adalah g(x), yang merupakan deret pangkat di mana koefisien dari (x') = [P(n, r)]/ (r!) = C(n, r). Jadi fungsi pembangkit eksponensial untuk P(n, r) adalah (1 + x)n, yang sama dengan fungsi pembangkit biasa dari C(n, r).

Contoh 2.3.3

Tentukan banyaknya cara menyusun 5 kelereng berjajar dengan menggunakan kelereng tiga warna (merah, biru, dan putih) sehingga setiap susunan memiliki paling sedikit satu kelereng masing-masing warna dengan asumsi kita mempunyai paling banyak 3 merah, paling banyak 2 putih , dan paling banyak 2 kelereng biru yang kita miliki. (Perhatikan perbedaan antara contoh ini dan Contoh 2.3.1.)

Bukti:

Dalam hal ini jumlah pengaturan akan menjadi koefisien x5/5! dalam $g(x) = (x + x^2/2! + x^3/3!)(x + x^2/2!)^2$ dan koefisien ini sama dengan koefisien $x^5/5!$ Di

$$h(x) = \left(x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots\right)\left(x + \frac{x^2}{2!}\right)^2$$
$$= (e^x - 1)\left(x + \frac{x^2}{2!}\right)^2$$

Analisis dalam Contoh 3.3.1 dan 3.3.3 digeneralisasikan sebagai berikut:

TEOREMA 3.3.1

Misalkan ada k jenis objek.

(a) Jika ada persediaan objek yang tidak terbatas pada masing-masing tipe ini, maka banyaknya permutasi r (r = 1, 2, . .) yang menggunakan objek dari tipe k ini adalah koefisien dari xr/r! dalam fungsi pembangkit eksponensial

$$g(x) = \left(1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots\right)^k = e^{kx}$$

(b) Jika persediaan benda dalam tipe i paling banyak ni, (di mana i = 1, 2, + ⋯+ , k), banyaknya permutasi r akan menjadi koefisien $x^r/r!$ di

$$h(x) = \left(1 + x + \frac{x^2}{2!} + \cdots + \frac{x^{n_1}}{n_1!}\right)\left((1 + x + \frac{x^2}{2!} + \cdots + \frac{x^{n_2}}{n_2!}\right) \cdots\cdots \left(1 + x + \frac{x^2}{2!} + \cdots + \frac{x^{n_k}}{n_k!}\right)$$

(c) (n!). S(r, n) = koefisien $x^r/r!$ di $(ex – 1)^n$ di mana S(r, n) adalah bilangan Stirling jenis kedua yang didefinisikan dalam Bab 1.

Contoh 3.3.4

(a) Tentukan banyaknya r-permutasi yang dapat dibentuk dengan menggunakan huruf I, M, S, dan P, di mana r adalah bilangan bulat positif.

(b) Tentukan banyaknya r-permutasi yang dapat dibentuk dengan menggunakan huruf-huruf yang muncul pada kata MISSISSIPPI sehingga banyaknya huruf yang muncul dalam suatu permutasi paling banyak sama dengan jumlah kemunculan huruf pada kata tersebut .

Bukti:

(a) Banyaknya permutasi-r adalah koefisien dari $x^r/r!$ dalam fungsi pembangkit eksponensial $g(x) = e^{4x}$, dan koefisien ini adalah 4^r.

(b) Dalam kata tersebut, frekuensi hurufnya adalah 4, 1, 2, dan 4. Jadi banyaknya permutasi r adalah koefisien dari $x^r/r!$ di

$$h(x) = \left(1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!}\right)(1 + x) \cdot \left(1 + x + \frac{x^2}{2!}\right)\left(1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!}\right)$$

di mana r paling banyak 11, jumlah frekuensi.

Contoh 3.3.5

Tentukan banyak cara untuk menampung 9 orang dalam 4 ruangan sehingga tidak ada ruangan yang kosong.

Bukti:

Jika x menunjukkan jumlah orang yang ditempatkan di sebuah ruangan, maka x adalah setidaknya satu dan paling banyak 6, dan ada 4 kamar. Jadi fungsi pembangkit eksponensial untuk masalah kombinatorial ini adalah:

$$g(x) = \left(x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^6}{6!}\right)^4$$

Banyaknya cara menampung 9 orang dalam 4 ruangan adalah koefisien $(x^9)/(9!)$ dalam g(x) dan koefisien ini sama dengan koefisien $(x^9)/(9!)$ dalam:

$$h(x) = (e^x - 1)4 = e^{4x} - 4e^{3x} + 6e^{2r} - 4e^x + 1$$

Jadi banyaknya susunan adalah $4^9 - (4)3^9 + (6)2^9 - 4$. [Perhatikan bahwa banyaknya susunan sama dengan (4!)S(9, 4) di mana S(9, 4) adalah Nomor stirling jenis kedua yang ditentukan dalam Bab 1.]

3.4 CATATAN DAN REFERENSI

Perlakuan komprehensif pertama dari fungsi pembangkit diberikan oleh Pierre Simon Marquis de Laplace (1749-1827). Tetapi metode pemecahan masalah dengan menggunakan fungsi pembangkit berasal dari karya Abraham de Moivre (1667-1754). Baik Leonhard Euler (1707-1783) dan Nicholas Bernoulli (1687-1759) juga menggunakan teknik ini dalam penyelidikan masalah kombinatorial tertentu: Euler tertarik, antara lain, dalam masalah partisi dan Bernoulli tertarik pada masalah derangement. Untuk pembahasan mendalam tentang fungsi pembangkit, lihat buku tentang kombinatorik oleh MacMahon (1960) atau Riordan (1958). Lihat juga Riordan (1964). Referensi umum lainnya termasuk bab-bab yang relevan dalam buku-buku oleh Cohen (1978), Krishnamurthy (1986), Liu (1968), Liu (1985), Roberts (1984), Tucker (1984), dan Townsend (1987).

3.5 LATIHAN

3.1 Temukan fungsi pembangkit biasa untuk barisan berikut.

(a) {1, 1, 1, 1, 0, 0, . . .} (b) {0, 0, 0, 0, 1, 1, . . .}

(c) {1, 1, 1, 1, . . .} (d) (1, -1, 1, -1, . . }

3.2 Temukan fungsi pembangkit biasa untuk barisan berikut.

(a) {1, 2, 3, 4, . . .} (b) {1, –2, 3, –4, . . .}

3.3 Tentukan barisan yang sesuai dengan fungsi pembangkit biasa berikut,

(a) $(2 + x)^4$ (b) $x^2 + e^x$

(c) $x^3(1 - x)^{-1}$

3.4 Temukan koefisien x^7 dalam $(1 - x)^k$ ketika k = 9 dan k = –9.

3.5 Temukan koefisien x^7 dalam $(1 + x)^k$ ketika k = 9 dan k = –9.

3.6 Temukan koefisien x^{23} dalam $(x^3 + x^4 + . . .)^5$.

3.7 Temukan fungsi pembangkit biasa f(x) yang dapat dikaitkan dengan masalah kombinatorial untuk menemukan jumlah solusi dalam bilangan bulat positif dari persamaan a + b + c + d = r.

3.8 Temukan fungsi pembangkit biasa yang terkait dengan masalah menemukan jumlah solusi dalam bilangan bulat nonnegatif dari persamaan 3a + 2b + 4c + 2d = r.

3.9 Tentukan banyaknya penyelesaian bilangan bulat dari persamaan p + q + r + s = 27 dimana setiap variabel paling sedikit 3 dan paling banyak 8.

3.10 Tentukan banyaknya solusi dari $x_1 + x_2 + \cdots + x_n = r$ di mana setiap variabel bernilai 0 atau 1.

3.11 Jika tiga buah dadu berbeda (bertanda A, B, dan C) dilempar, tentukan banyak cara untuk mendapatkan total 13 buah dadu.

3.12 Selesaikan Soal 2.11 jika dadu pertama (bertanda A) menunjukkan bilangan genap.

3.13 Temukan banyak cara untuk mengalokasikan 9 objek identik ke 3 lokasi berbeda (dinomori sebagai pertama, kedua, dan ketiga) sehingga setiap lokasi mendapatkan setidaknya satu objek dan lokasi ketiga tidak mendapatkan lebih dari 3 objek.

3.14 Tentukan fungsi pembangkitan biasa yang berhubungan dengan masalah kombinatorial memilih 9 kelereng dari sebuah kantong yang memiliki 3 kelereng merah identik, 4 kelereng biru identik, dan 5 kelereng hijau identik sehingga dalam setiap pilihan semua warna diwakili dan tidak ada warna yang memiliki mayoritas mutlak.

3.15 Buktikan: $(1 + x^m)^n = 1 + C(n, 1)x^m + C(n, 2)(x^m)^2 + \cdots + (x^m)^n$.

3.16 Tentukan banyaknya solusi bilangan bulat dari persamaan a + b + c + d + e + f = 20, di mana a paling sedikit 1 dan paling banyak 5 dan variabel lainnya paling sedikit 2 dengan (a) metode yang dikembangkan di Bab 1, dan (b) mempertimbangkan fungsi pembangkit yang sesuai.

3.17 Ada 10 kotak hadiah yang identik. Setiap kotak harus dibungkus dengan kertas pembungkus merah atau biru atau hijau atau kuning. Kertas merah yang tersedia dapat digunakan untuk membungkus paling banyak 2 kotak dan kertas biru yang tersedia dapat digunakan untuk membungkus paling banyak 3 kotak. Tuliskan fungsi pembangkit biasa yang terkait dengan masalah menemukan banyak cara untuk membungkus 10 kotak ini.

3.18 Ada 9 orang dalam satu kelompok. Tentukan banyak cara untuk mengumpulkan Rp 135.000 dari grup ini jika pemimpin kelompok akan memberikan setidaknya Rp 15.000 dan paling banyak Rp 30.000 dan setiap anggota lainnya akan memberikan paling banyak Rp 15.000.

3.19 Temukan fungsi pembangkit biasa yang terkait dengan masalah menemukan jumlah solusi dalam bilangan bulat dari pertidaksamaan $a + b + c \leq r$, di mana setiap variabel paling sedikit 2 dan paling banyak 5.

3.20 Seorang peserta dalam kontes dinilai pada skala 1 sampai 6 oleh masing-masing dari 4 juri. Untuk menjadi finalis, seorang peserta harus mendapatkan nilai minimal 22. Tentukan banyaknya cara juri menilai seorang peserta sehingga dia dapat menjadi finalis.

3.21 Sebuah kelompok ekspedisi Antartika terdiri dari ilmuwan yang mewakili Amerika Serikat, Uni Soviet, dan Inggris. Tentukan banyak cara untuk membentuk kelompok yang terdiri dari 9 ilmuwan sehingga tidak satu pun dari ketiga negara tersebut memiliki mayoritas mutlak dalam kelompok tersebut.

3.22 Gunakan argumen kombinatorial untuk membuktikan bahwa koefisien x^{2n+1} di f(x) sama dengan koefisien x^{2n-2} di g(x), di mana $f(x) = (1 + x + \cdots + x^n)^3$ dan $g(x) = (1 + x + x^2 + \cdots + x^{n-1})^3$. Temukan koefisien ini.

3.23 Tentukan banyak cara membagikan 8 apel dan 6 jeruk kepada 3 anak sehingga setiap anak dapat memperoleh paling sedikit 2 apel dan paling banyak 2 jeruk.

3.24 Buktikan (a) $1 + 2 + 3 + \cdots + r = [r(r + 1)]/2$ dan (b) $1 + 2^2 + 3^2 + \cdots + r^2 = [r(r + 1)(2r + 1)]/6$.

3.25 Tentukan banyak cara untuk menyimpan p kelereng merah identik dalam m kotak di satu rak dan q kelereng biru identik dalam n kotak di rak lain sehingga tidak ada kotak yang kosong. (Karena tidak ada kotak yang kosong p tidak boleh kurang dari m dan q tidak boleh kurang dari r.) Selesaikan masalah ketika p = 6, q = 7, m = 3, dan n = 4.

3.26 Temukan fungsi pembangkit biasa untuk barisan:

(a) $\{a_r\}$ di mana $a_r = k^r$, di mana k adalah konstanta

(b) $\{b_r\}$ di mana $b_r = rK^r$

(c) $\{c_r\}$ di mana $c_r = k + 2k^2 + 3k^3 + \cdots + rk^r$

3.27 Jumlah empat bilangan bulat positif dalam urutan tidak menurun adalah r dan ar adalah banyak cara untuk memilih empat bilangan bulat ini. Temukan fungsi pembangkit biasa yang terkait dengan barisan {ar}.

3.28 Jika X suatu himpunan dengan n anggota, tunjukkan bahwa banyaknya himpunan bagian dari X dengan (r – 1) anggota sama dengan banyaknya penyelesaian persamaan $y_1, + y_2 + \bullet \bullet \bullet + yr = (n - 1)$, di mana dua variabel pertama tidak negatif dan variabel lainnya positif.

3.29 Misal X = {1, 2, 3, + ⋯+ , n}. Tentukan banyaknya himpunan bagian dari X sedemikian sehingga setiap himpunan bagian memiliki r elemen dan tidak ada dua elemen dalam suatu himpunan yang merupakan bilangan bulat berurutan.

3.30 Jika r adalah bilangan bulat positif, partisi dari r adalah kumpulan bilangan bulat positif yang jumlahnya r. Sebuah partisi dikatakan berbeda jika bilangan bulat di dalamnya berbeda. Misalnya {3, 1} adalah partisi 4 yang berbeda, sedangkan {2, 2} adalah partisi 4 yang tidak berbeda. Banyaknya partisi dari r dilambangkan dengan p(r) dan banyaknya partisi yang berbeda dari r dilambangkan dengan pd(r). Dapatkan fungsi pembangkit untuk menghitung p(r) dan pd(r).

3.31 Tunjukkan bahwa jumlah partisi berbeda dari bilangan bulat positif r sama dengan jumlah partisi r menjadi bilangan bulat positif ganjil.

3.32 Misalkan p(r; n) adalah banyaknya partisi dari r sedemikian sehingga pada setiap partisi tidak ada elemen yang melebihi n. Tunjukkan bahwa p(r; r) = p(r).

3.33 Tunjukkan bahwa setiap bilangan bulat nonnegatif dapat ditulis secara unik dalam bentuk biner.

3.34 Temukan fungsi pembangkit eksponensial yang terkait dengan barisan berikut.

(a) {1, 1, 1, 1, 0, 0, 0, 0, . . .} (b) {0, 0, 0, 0, 1, 1, 1, 1, . . .}

(c) {1, 2, 22, 23, 24 . . .} (d) {1, 1, 2 . 2, 3 . 22, 4.23, . . .}

3.35 (a) Gunakan argumen kombinatorial untuk membuktikan bahwa (ex)n = enx.

(b) Buktikan bahwa $e^x + e^{-x} = 2(1 + x^2/2! + x^4/4! + \cdots)$.

(c) Buktikan bahwa $e^x - e^{-x} = 2(x + x^3/3! + x^5/5! + \cdots)$.

3.36 Misal X = {A, B, C, D}. Dengan menggunakan fungsi pembangkit eksponensial, dapatkan (a) jumlah permutasi r yang dapat dibentuk dengan menggunakan empat huruf ini sehingga dalam setiap permutasi paling sedikit ada satu A, paling sedikit satu B, dan paling sedikit satu C, dan (b) banyaknya permutasi r yang dapat dibentuk sedemikian rupa sehingga pada setiap permutasi terdapat bilangan genap A dan bilangan ganjil B.

3.37 Tentukan banyaknya bilangan biner r-digit yang dapat dibentuk menggunakan bilangan genap 0 dan bilangan genap l.

3.38 (a) Tentukan banyak cara kantor pusat sebuah perusahaan dapat mengalokasikan sembilan komputer baru yang identik ke empat kantor cabang yang berbeda sehingga setiap kantor mendapatkan setidaknya satu komputer baru.

(b) Tentukan banyak cara kantor pusat sebuah perusahaan dapat mengalokasikan sembilan karyawan baru ke empat kantor cabang yang berbeda sehingga setiap kantor mendapatkan setidaknya satu karyawan baru.

3.39 Tentukan banyaknya (a) permutasi huruf-huruf yang muncul pada kata MISSISSIPPI, (b) banyaknya 6- permutasi huruf yang muncul pada kata ini, dan (c) banyaknya 6-permutasi huruf dari kata ini sedemikian rupa sehingga dalam setiap permutasi setiap huruf dari kata muncul setidaknya sekali.

3.40 Tentukan barisan bilangan sembilan angka yang dapat dibentuk dengan menggunakan angka 0, 1, 2, dan 3 sedemikian rupa sehingga (a) setiap barisan memiliki bilangan genap 0, (b) setiap barisan memiliki bilangan ganjil 0, (c) setiap barisan memiliki bilangan genap 0 dan l ganjil, (d) jumlah total 0 dan l ganjil, dan (e) tidak ada angka yang muncul tepat dua kali.

3.41 Dapatkan fungsi pembangkit yang sesuai dengan masalah kombinatorial untuk menemukan jumlah kata sandi dengan panjang r dari alfabet yang terdiri dari lima huruf berbeda sehingga di setiap kata kode setiap huruf alfabet muncul setidaknya satu kali dan huruf pertama muncul sejumlah genap waktu.

BAB 4
HUBUNGAN PERULANGAN

4.1 PENDAHULUAN

Pertimbangkan barisan a_0, a_u a_2, . . . , di mana ar adalah solusi dari masalah kombinatorial tertentu yang bergantung pada input r. Dalam Bab 2 kita membahas beberapa metode untuk menghitung ar menggunakan fungsi pembangkit. Dalam beberapa kasus, akan mungkin untuk mereduksi perhitungan suku ke-r dari barisan menjadi anggota barisan yang lebih awal jika ar dapat dinyatakan sebagai fungsi dari elemen barisan yang lebih awal. Sebagai contoh, perhatikan barisan deret aritmatika 4, 7, 10, 13, 16, . . . , di mana bilangan awal a_0 adalah 4 dan selisih d adalah 3. Maka suku ke-r dari barisan tersebut dapat dinyatakan dalam suku ke (r – l) dengan persamaan a_r = a_{r-1}, + d. Persamaan ini adalah contoh dari relasi rekurensi. Kondisi a_0 = 4 disebut kondisi awal dari relasi ini. Jelas, setelah kondisi awal dan perbedaan umum diketahui, suku sembarang dapat diperoleh dengan menghitung a^1, a^2, . . . , secara berurutan. Atau kita dapat memperoleh suku ke-r dengan menyelesaikan relasi perulangan. Dalam hal ini solusinya adalah ar = 4 + 3^r, di mana r adalah sembarang bilangan bulat tak negatif. Demikian pula, jika kita mengambil barisan deret geometri 4, 4.3, 4.3^2, 4.3^3 . . . , relasi perulangannya adalah ar = 3 . a_{r-1}, dengan kondisi awal a_0 = 4 dan penyelesaiannya adalah ar = 4 . 3^r.

Hubungan Perulangan dan Persamaan Selisih

Selisih pertama d(a_n) dari barisan {an} bilangan real adalah selisih a_n –a_{n-1}. Selisih kedua d2(an) adalah d(an) – d(an–1), yang sama dengan – 2a_{n-1} + a_{n-2}. Lebih umum perbedaan ke-k dk (an) adalah dk-1(an) – dk-1(an_x) . Persamaan diferensial adalah persamaan yang melibatkan an dan perbedaannya. Misalnya, 3d2(an) + 2d(an) + 7an = 0 adalah persamaan beda homogen orde dua. Amati bahwa setiap ai, (i = 0, 1, 2, . . ,n – 1) dapat dinyatakan dalam an dan perbedaan ini karena an-1 = an – d(an);

$$a_{n-2} = a_{n-1} - d(a_{n-1}) = [a_n - d(a_n)] - d(a_n - d(a_n))$$
$$= a_n - 2d(a_n) + d^2(a_n)$$

dan seterusnya. Dengan demikian setiap relasi perulangan dapat dirumuskan sebagai persamaan beda. Di sisi lain, dengan menggunakan definisi perbedaan ini, persamaan perbedaan apa pun dapat dirumuskan sebagai hubungan perulangan. Misalnya, persamaan selisih 3d2(an) + 2d(an) + 7an = 0 dapat dinyatakan sebagai relasi perulangan 12an = 8an–1 – 3an–2. Jadi beberapa penulis menggunakan istilah persamaan perbedaan dan hubungan perulangan secara bergantian. Metode untuk memecahkan hubungan perulangan dikembangkan awalnya menggunakan teknik yang digunakan untuk memecahkan persamaan perbedaan. Persamaan diferensial biasanya digunakan untuk memperkirakan persamaan diferensial ketika menyelesaikan persamaan diferensial menggunakan komputer.

Perhatikan bahwa dalam relasi tipe ar = ar–1, + ar–2 kita perlu mengetahui a0 dan a1, untuk memperoleh sembarang ar (r > 1), dan oleh karena itu kita memerlukan dua kondisi awal untuk menyelesaikan persamaan ini. Jadi dengan informasi yang tersedia dalam himpunan kondisi awal dari hubungan perulangan yang diberikan dalam banyak kasus, seseorang harus dapat menghitung secara berurutan setiap suku arbitrer dari urutan. Kita akan mempelajari teknik-teknik untuk memecahkan jenis-jenis relasi perulangan tertentu nanti di bagian ini. Tidak ada metode umum untuk menyelesaikan semua relasi perulangan.

Contoh 4.1.1

Relasi perulangan ar = rar–1, dengan kondisi awal a0 = 1 memiliki solusi ar = r! (r = 1, 2, . . .).

Contoh 4.1.2

Temukan relasi perulangan untuk mendapatkan an, banyaknya cara menyusun n elemen berbeda dalam suatu baris.

Bukti:

Ada n cara memilih elemen untuk ditempatkan pada posisi pertama baris. Setelah menempatkan elemen di posisi pertama, banyak cara untuk mengatur elemen yang tersisa (n – 1) adalah an-1. Jadi kita memiliki relasi rekurensi an = nan-1 dengan kondisi awal a1, = 1, solusinya (dengan contoh sebelumnya) adalah = n!

Contoh 3.1.3

Misalkan tingkat bunga yang ditawarkan oleh bank kepada deposan adalah r% per tahun. Jika an adalah jumlah deposito pada akhir n tahun, dapatkan hubungan perulangan untuk a jika (a) bunganya sederhana, dan (b) bunganya dimajemukkan setiap tahun.

Bukti:

(a) Jika a0 adalah setoran awal, pada akhir tahun k, ra0 ditambahkan ke ak jika bunganya sederhana. Jadi relasi perulangannya adalah ak + 1= ak + ra0, dimana k = 0, 1, 2, Dengan iterasi kita lihat bahwa solusinya adalah:

ak +1 = ak–1 + ra0 + ra0 = ak–2 + 3ra0 = = a0 + (k + 1)ra0

Jadi solusi dari relasi perulangan adalah an = (1 + nr)a0.

(b) Jika bunga dimajemukkan setiap tahun, relasi perulangannya adalah ak +1= ak + rak = (1 + r)ak , penyelesaiannya, dengan iterasi, adalah an = (1 + r)na0.

Contoh 4.1.4

Temukan relasi perulangan untuk barisan Fibonacci 1, 2, 3, 5, 8, 13, di mana suku ke-r adalah jumlah suku ke-(r – ke-l) dan suku ke-(r – 2). Jelas, relasinya adalah an – an–1, – an–2 = 0 dengan kondisi awal a1 = 1 dan a2 = 2 . (Bilangan yang muncul dalam barisan disebut bilangan Fibonacci dan muncul di banyak bidang matematika kombinatorial. Kita akan membahas teknik penyelesaian nanti.)

Rekursi dan Pengulangan

Relasi perulangan seperti yang kita lihat di sini adalah rumus rekursif (lihat Bagian 3 dari Bab 0) untuk menghitung jumlah cara untuk melakukan prosedur yang melibatkan n objek dalam hal jumlah cara untuk melakukannya dengan objek yang lebih sedikit. Penalaran rekursif yang terlibat dalam membangun model hubungan pengulangan dari masalah penghitungan adalah logika yang sama yang digunakan dalam merancang subrutin komputer rekursif yang memanggil diri mereka sendiri. Ide dasar dari setiap prosedur rekursif dalam ilmu komputer adalah bahwa ia memanggil dirinya sendiri untuk memecahkan masalah dengan memecahkan masalah serupa yang lebih kecil dari masalah aslinya. Fitur penting dari rekursi adalah konsep bekerja mundur. Namun, kita tidak dapat menggunakan program rekursif untuk menghitung nilai dalam relasi rekursif karena relasi rekursif dimaksudkan untuk tabulasi nilai ke depan, bukan untuk komputasi rekursif mundur.

4.2 HUBUNGAN PERULANGAN HOMOGEN

Seperti disebutkan sebelumnya, tidak ada metode umum solusi untuk relasi perulangan arbitrer. Berikut ini kita mempelajari kelas luas dari hubungan perulangan yang teknik penyelesaiannya diketahui.

TEOREMA 4.2.1

Jika ci (i = 1, 2, . . , r) adalah konstanta, relasi perulangan dari bentuk an = c1an–1 + c2an–2 + + cran–r + f(n) adalah disebut relasi rekurensi linier dengan koefisien konstanta orde r. Relasi perulangan adalah homogen jika fungsi f(n) = 0. Jika g(n) adalah fungsi sedemikian sehingga an = g(n) untuk n = 0, 1, 2, , maka g(n) adalah solusi dari kekambuhan hubungan.

Contoh 4.2.1

Dapat dibuktikan dengan substitusi bahwa g(n) = A . 2n + B. n. 2n + n2 . 2n-1 (di mana A dan B adalah konstanta arbitrer) adalah solusi dari persamaan perulangan linier tak homogen orde kedua berikut dengan koefisien konstan: an = 4an- 1 – 4an–2 + 2n. (Kami akan mempelajari teknik solusi untuk mendapatkan solusi umum seperti itu di bagian ini.)

TEOREMA 4.2.1 (Prinsip Superposisi)

Jika gi(n) , di mana i = 1, 2, . . . , adalah solusi dari:

an = c1an–1 + c2an–2 + • • • + cran–r + fi(n)

maka kombinasi linier dari k solusi bentuk A1g1(n) + A2g2(n) + • • • + Akgk (n) adalah solusi dari relasi perulangan an = c1an–1 + c2an–2 + • • • + cran–r + A1f1(n) + • • • + Akfk (n) di mana Ai (i = 1, 2, . . , k) adalah bilangan real. Secara khusus, setiap kombinasi linear dari solusi dari relasi rekurensi homogen sekali lagi merupakan solusi dari relasi rekurensi homogen.

Bukti :

Membiarkan

Karena gi(n) adalah solusi dari:

h(n) = A1g1(n) + A2g2(n) + • • • + Akgk (n)

an = c1an–1 + c2an–2 + • • • + cran–r + fi(n)

kita punya:

gi(n) = c1gi(n – 1) + c2gi(n –2) + • • • + crgi(n – r) + fi(n)

dan maka dari itu:

$$h(n) = c1h(n - 1) + c2h(n - 2) + \bullet \bullet \bullet + crh(n - r) + A1f1(n) + \bullet \bullet \bullet + Akfk\ (n)$$

yang membuktikan pernyataan kami.

Ada teknik sederhana untuk menyelesaikan hubungan rekurensi linier homogen dengan koefisien konstan. Misalkan ar = xr menjadi solusi dari relasi:

$$an = c1an-1 + c2an-2 + \bullet \bullet \bullet + cran-r$$

Kemudian:

$$xn = c1xn-1 + c2xn-2 + \bullet \bullet \bullet + crxn\text{-}r$$

Jika kita mengabaikan solusi trivial x = 0 kita mendapatkan persamaan polynomial:

$$xr - c1xr-1 - c2xr-2 - \bullet \bullet \bullet - cr = 0$$

Persamaan polinomial derajat r ini disebut persamaan karakteristik dari hubungan perulangan yang memiliki akar r secara umum. Sangat mungkin bahwa persamaan memiliki akar ganda atau beberapa akar kompleks.

Jika xi (i = 1, 2, . . , r) adalah akar-akar r dari persamaan karakteristik maka an = (xi)n jelas merupakan solusi dari hubungan perulangan homogen dan oleh karena itu dengan proposisi sebelumnya setiap kombinasi linier dari solusi tersebut juga larutan. Sebagai contoh, relasi rekurensi homogen orde kedua an = 5an–1, – 6an–2 memiliki persamaan karakteristik x2 – 5x + 6 = 0 akar-akarnya adalah x1, = 2 dan x2 = 3. Jadi an = A(2)n + B(3)n, untuk sembarang pilihan konstanta sembarang A dan B, juga merupakan solusi dari relasi perulangan. Sebaliknya, jika semua r akar x, (i = 1,2, . . , r) nyata dan berbeda, dapat dibuktikan bahwa setiap solusi umum adalah kombinasi linier dari solusi ini (xi)n.

Konstanta arbitrer r yang muncul dalam solusi umum dapat dievaluasi dalam beberapa kasus memberikan solusi lengkap (tidak harus unik) untuk masalah jika r kondisi awal diketahui. Keberadaan dan keunikan solusi dipastikan jika r kondisi awal berurutan diketahui. Kami menyatakan hasil ini sebagai proposisi yang buktinya dihilangkan di sini.

TEOREMA 4.2.2

Jika akar-akar r x, (i = 1,2, . . , r) dari persamaan karakteristik dari suatu relasi rekurensi homogen linier orde-r nyata dan berbeda, setiap solusi umum dari relasi rekurensi adalah kombinasi linear dari solusi (xi)n. Selain itu, jika r berturut-turut nilai awal ak, ak +1 . . . , ak +r-1, dari relasi perulangan diketahui, solusi dapat diperoleh dengan mengevaluasi r konstanta arbitrer menggunakan r nilai awal berurutan ini dan solusi ini unik.

Contoh 4.2.2

Selesaikan sebuah – 9an–2 = 0, di mana:

(a) a0 = 6, a1, = 12

(b) a3 = 324, a4 = 486

(c) a0 = 6, a2 = 54

(d) a0 = 6, a2 = 10

Bukti:

Akar persamaan karakteristik r2 – 9 = 0 adalah 3 dan –3. Jadi, solusi umum dari relasi perulangan yang diberikan adalah dalam bentuk an = A(3)n + B(– 3)n, di mana A dan B adalah konstanta arbitrer yang dievaluasi menggunakan dua kondisi awal yang diberikan.

(a) Jika a0 = 6, maka A + B = 6 . Jika a1, = 12, maka 3A – 3B = 12. Memecahkan dua persamaan simultan ini di A dan B kita mendapatkan A = 5 dan B = 1, memberikan solusi unik (karena kondisi awal berurutan) untuk masalah an = 5 (3)n + (–3)n. Menempatkan n = 2, 3, 4, . . . , kita dapat menghitung a2 = 54, a3 = 324, dan seterusnya.

(b) Jika a3 = 324, maka 27A – 27B = 324. Jika a4 = 486, maka 81A + 81B = 486. Memecahkan kedua persamaan ini, kita dapatkan A = 9 dan B = –3, sekali lagi memberikan solusi unik.

(c) Kali ini dua kondisi awal yang diberikan tidak berurutan. a0 = 6 menyiratkan bahwa A + B = 6 dan a2 = 54 menyiratkan bahwa 9A + 9B = 54, memberikan satu persamaan A + B = 6 untuk menentukan dua konstanta. Misalnya, A = 2, B = 4 menghasilkan an = 2(3)n + 4(–3)n, yang mendefinisikan a0 = 6, a1 = –6, a2 = 54, a3 = –54, dan seterusnya. Tetapi A = 1, B = 5 menghasilkan an = (3)n + 5(–3)n, yang mendefinisikan a0 = 6, a1 = –12, a2 = 54, a3 = – 108. Jadi, penyelesaiannya tidak unik.

(d) Kondisi awal yang tidak berurutan menyiratkan bahwa A + B = 6 dan 9A + 9B = 10, menunjukkan bahwa tidak ada solusi.

Contoh 4.2.3

Selesaikan relasi perulangan Fibonacci an = an–1 + an–2 dengan kondisi awal berurutan a0 = 1 dan a1 = 1.

Bukti:

Persamaan karakteristik adalah x2 – x – 1 = 0, dua akarnya adalah dan . Jadi setiap solusi berbentuk an = A(x1)n + B(x2)n, di mana A dan B adalah konstanta arbitrer Dengan menggunakan kondisi awal kita peroleh dan . Dengan mengganti nilai-nilai konstanta arbitrer ini dalam solusi, kita mendapatkan solusi unik , di mana t adalah bilangan irasional , yang dikenal sebagai rasio emas.

Contoh 4.2.4

Tentukan banyaknya himpunan bagian dari suatu himpunan yang memiliki n anggota.

Bukti:

Biarkan himpunan menjadi X = {1, 2, 3, . . . , n} dan misalkan jumlah himpunan bagian dari X adalah an. Sekarang setiap himpunan bagian dari X termasuk ke dalam salah satu dari dua kelas A dan B berikut: (a) n bukan merupakan elemen di salah satu himpunan bagian di kelas A dan (b) n adalah elemen di setiap himpunan bagian di kelas B.

Dengan asumsi kita kelas A memiliki an-1, himpunan bagian dari X. Satu-satunya cara kita bisa mendapatkan himpunan bagian dari X yang berisi n adalah dengan menghubungkan n ke himpunan bagian dari kelas A. Jadi jumlah himpunan bagian di kelas B persis sama dengan banyaknya himpunan bagian di kelas A. Dengan kata lain, kita memiliki relasi perulangan an = 2an-1, dengan kondisi awal a0 = 1. Ingat bahwa himpunan kosong adalah himpunan bagian dari setiap himpunan. Persamaan karakteristik adalah x – 2 = 0, memberikan solusi unik an = 2n.

Contoh 4.2.5

Temukan solusi unik dari relasi perulangan an = 3an–1 + 4an–2 – 12an–3, di mana a0 = 2, a1 = 5, dan a2 = 13.

Bukti:

Akar persamaan karakteristik x3 – 3x2 – 4x + 12 = 0 adalah 2, – 2, dan 3. Jadi solusi umumnya adalah an = p(2)n + q(–2)n + r(3)n dan kondisi awal menyiratkan bahwa:

$$\begin{aligned} p + q + r &= 2 \\ 2p - 2q + 3r &= 5 \\ 4p + 4q + 9r &= 13 \end{aligned}$$

Pada penyelesaian sistem linier ini, kita mendapatkan p = 1, q = 0, dan r = 1. Jadi solusi uniknya adalah an = 2n + 3n.

Kami selanjutnya memeriksa kasus ketika persamaan karakteristik dari hubungan perulangan memiliki akar berulang (multiple). Perhatikan, misalnya, relasi perulangan an = 4an-1 – 4an–2, yang memiliki persamaan karakteristik (x – |2)2 = 0 dengan akar x1 = 2 dan x2 = 2. Dengan kata lain, 2 adalah pengulangan akar multiplisitas 2. Tentu saja, A(2)n adalah solusi dari relasi perulangan, tetapi setiap solusi tidak harus berbentuk ini. Dapat dengan mudah diverifikasi bahwa B . n . (2)n juga merupakan solusi, jadi dengan prinsip superposisi A(2)n + B . n.(2)n juga merupakan solusi untuk sembarang A dan B. Ternyata setiap solusi umum dari relasi perulangan berbentuk ini. Fakta bahwa A(2)n sendiri tidak dapat menjadi solusi umum juga jelas karena jika kita mempertimbangkan dua kondisi awal yang berurutan, katakanlah a0 = 1 dan a1 = 4, kita melihat bahwa A = 1 dan juga A = 2.

Ketika persamaan karakteristik memiliki akar ganda, hasil berikut adalah generalisasi dari teorema sebelumnya.

Bukti dihilangkan di sini. Untuk detailnya, lihat Roberts (1984).

TEOREMA 4.2.3

(a) Misalkan (x – t)s adalah faktor dari persamaan karakteristik sehingga t adalah akar dari multiplisitas s. Kemudian u = (t)n(A1 + A2n + A3n2 + • • •+ Asns–1) adalah solusi dari relasi perulangan di mana Aj (j = 1, 2, . . , s) adalah konstanta arbitrer. Solusi u ini disebut solusi dasar dari relasi terhadap akar t.

(b) Misalkan akar dari relasi perulangan adalah tk , di mana multiplisitas tk adalah sk (k= 1, 2, . . , q) , dan anggaplah uk adalah solusi dasar terhadap akar tk . Maka setiap solusi dari relasi perulangan adalah jumlah dari q solusi dasar tersebut.

Contoh 4.2.6

Tentukan solusi umum dari relasi perulangan yang akar-akarnya adalah 2, 2, 2, –3, 4, dan 4.

Bukti:

Solusi dasar untuk akar berulang 2 adalah u1 , = 2n(A1 + A2n + A3n2) . Solusi dasar untuk akar – 3 adalah u2 = A4(–3)n. Solusi dasar untuk akar berulang 4 adalah u3 = 4n(A5 + A6n). Jadi solusi umumnya adalah an = u1 + u2 + u3.

4.3 HUBUNGAN PERULANGAN TIDAK HOMOGEN

Kami sekarang melanjutkan untuk menganalisis hubungan perulangan linier dengan koefisien konstan tipe an = hn + f(n) , di mana hn = c1an-1 + c2an–2 + • • + cran–r dan f(n) adalah fungsi dari n. Di sini relasi an = hn disebut bagian homogen dari relasi tak homogen yang diberikan. Jika an = un adalah solusi bagian homogen dan jika an = vn adalah solusi dari relasi tak homogen yang diberikan, maka dengan prinsip superposisi kita tahu bahwa an = un + vn juga merupakan solusi dari relasi tak homogen yang sama. Jika un memiliki r konstanta arbitrer, maka un + vn juga memiliki r konstanta arbitrer. Jika r kondisi awal berurutan dari hubungan tak homogen diketahui, kondisi awal ini dapat digunakan untuk mendefinisikan sistem linear dari r persamaan dalam r variabel yang memberikan solusi unik. Dengan kata lain, jika un adalah solusi umum dari bagian homogen dari relasi perulangan tak homogen, dan jika vn adalah solusi partikular dari relasi tak homogen, maka un + vn adalah solusi umum relasi tak homogen yang sama.

Contoh 4.3.1

Temukan solusi umum dari an = 5an–1 – 6an–2 + 6(4)n.

Bukti:

Penyelesaian bagian homogen adalah un = A(2)n + B(3)n, di mana A dan B adalah konstanta sembarang. Dapat dibuktikan bahwa vn = (48)(4)n adalah solusi khusus dari relasi tak homogen yang diberikan. Oleh karena itu, solusi umum dari relasi tersebut adalah an = un + vn.

Berbeda dengan kasus homogen, tidak ada metode umum untuk mendapatkan solusi khusus untuk masalah homogen yang arbitrer. Namun, ada teknik yang tersedia untuk kasus khusus tertentu. Kami memiliki dua kasus khusus seperti: (l)f(n) = nk , di mana k adalah bilangan bulat non-negatif, dan (2)f(n) = (q)n, di mana q adalah bilangan rasional yang tidak sama dengan 1. prinsip superposisi disebut jika ada kombinasi linier fungsi dari kedua jenis ini. Teknik-teknik tersebut adalah sebagai berikut:

1. Jika f(n) = c(q)n (di mana c adalah konstanta yang diketahui) dan jika q bukan akar persamaan karakteristik, maka pilihan untuk solusi tertentu adalah A(q)n, di mana A adalah konstanta yang akan dievaluasi dengan mensubstitusikan = A(q)n ke dalam persamaan tak homogen. Jika q adalah akar dari persamaan karakteristik dengan multiplisitas k, pilihan untuk solusi tertentu adalah A(n)k (q)n.
2. Jika f(n) = c(n)k dan jika 1 bukan akar dari persamaan karakteristik, polinomial dalam n derajat k berbentuk A0 + A1n + A2n2 + • • • + Aknk adalah pilihan untuk solusi tertentu. Jika 1 adalah akar dari multiplisitas t, A0nt + A1nt+1 + • • • + Aknt+k adalah pilihannya.

Contoh 4.3.2

Jika persamaan karakteristik dari relasi rekurensi tak homogen tertentu adalah (x – 1)2(x – 2)(x – 3)2 = 0, tentukan pilihan untuk solusi tertentu ketika

(a) f(n) = 4n3 + 5n

(b) f(n) = 4n

(c) f(n) = 3n

Bukti:

Akar persamaan karakteristik adalah 1 (dengan multiplisitas 2), 2 (dengan multiplisitas 1), dan 3 (dengan multiplisitas 2). Biarkan un menjadi solusi umum dari bagian homogen dan vn menjadi pilihan solusi tertentu. Kemudian:

$$u_n = c_1 + c_2 n + c_3 2^n + c_4 3^n + c_5 \cdot n \cdot 3^n$$

(a) $v_n = An^2 + Bn^3 + Cn^4 + Dn^5$
(b) $v_n = A \cdot 4^n$
(c) $v_n = A \cdot n^2 \cdot 3^n$

Contoh 4.3.3
Diskusikan solusi dari an = kan–1 + f(n) , di mana k adalah konstanta.
Bukti:
Seperti sebelumnya, an = un + vn, di mana un adalah solusi bagian homogen dan vn adalah solusi partikular.

Kasus (1): k = 1. un = c, di mana c adalah konstanta sembarang, jadi an = c + vn, di mana sifat vn bergantung pada f(n) dan fakta bahwa un adalah konstanta. Namun:

$$\begin{aligned} a_n &= a_{n-1} + f(n) \\ a_{n-1} &= a_{n-2} + f(n-1) \\ &\vdots \\ a_2 &= a_1 + f(2) \\ a_1 &= a_0 + f(1) \end{aligned}$$

Menambahkan n persamaan ini, kita mendapatkan:

an = a0 +f(1) + f(2) + • • • +f(n)

Dengan demikian:

f(l) + f(2) + • • • + f(n) = an – a0 = c + vn – a0

dimana nilai c dapat dievaluasi dengan menggunakan kondisi awal a0.

Kasus (2): k tidak sama dengan 1. un = c . kn dan seperti sebelumnya, vn bergantung pada f(n) dan un.
Contoh 4.3.4
Hitung jumlah kuadrat dari n bilangan bulat positif pertama.

Bukti:

Tulis f(n) = n2. Kita harus menghitung f(l) + f(2) + • • • + f(n) . Pertimbangkan hubungan perulangan an = an-1 + n2 dengan a0 = 0. Bagian homogen memberikan solusi un = c. Pilihan untuk solusi tertentu adalah vn = An + Bn2 + Cn3. Jadi an = c + An + Bn2 + Cn3. Kondisi awal menyiratkan bahwa c = 0. Mengganti an dalam relasi perulangan, kita mendapatkan:

An + Bn2 + Cn3 = A(n – 1) + B(n – l)2 + C(n – l)3 + n2

Menyamakan koefisien n, n2 dan suku konstan di kedua sisi kita dapatkan $A = \frac{1}{6}, B = \frac{1}{2}$ dan dengan demikian:

$$a_n = \frac{1}{6}n + \frac{1}{2}n^2 + \frac{1}{3}n^3 = \frac{n(n+1)(2n+1)}{6}$$

yang tentu saja sama dengan jumlah kuadrat dari n bilangan asli pertama.

Contoh 4.3.5

Selesaikan an = an–1 + 12n2, di mana a0 = 5.

Bukti:

Kita melihat an = a0 + (12)(n)(n + 1)(2n + 1)/6 dari Contoh 3.3.4. Jadi, an = 5 + 2n(n + 1)(2n + 1).

4.4 HUBUNGAN PERULANGAN DAN FUNGSI PEMBANGKIT

Dalam banyak contoh, suku ke-n an dalam relasi perulangan dapat diperoleh sebagai koefisien xn dalam perluasan deret pangkat dari suatu fungsi g(x) yang dapat dianggap sebagai fungsi pembangkit untuk relasi perulangan yang diberikan. Cukup sering persamaan fungsional untuk g(x) dapat diselesaikan secara aljabar dan kemudian diperoleh dengan menyatakan g(x) sebagai deret pangkat. Dengan kata lain, relasi perulangan diselesaikan dengan menggunakan fungsi pembangkit terkait.

Contoh 3.4.1

Selesaikan relasi perulangan an = 2an-1 dengan menggunakan fungsi pembangkit terkait.

Bukti:

Misalkan g(x) = a0 + a1x + • • • + anxn + • • • menjadi fungsi pembangkit yang terkait. Mengalikan kedua sisi dari relasi perulangan dengan xn, kita mendapatkan anxn = 2an–1xn, di mana n = 0, 1, 2, Jadi:

$$\begin{aligned} a_1x &= 2a_0x \\ a_2x^2 &= 2a_1x^2 \\ &\vdots \\ a_nx^n &= 2a_{n-1}x^n \end{aligned}$$

Pada penjumlahan kita melihat bahwa g(x) – a0 = 2xg(x) , yang merupakan persamaan fungsional untuk g(x) , dan persamaan ini dapat diselesaikan untuk g(x) . Kami mendapatkan g(x) = a0/(1 – 2x) . Jadi an = a02n, yang merupakan koefisien xn dalam ekspansi deret pangkat g(x) .

Contoh 4.4.2

Selesaikan an = 2an-1 – (n/3), di mana a0 = 1.

Bukti:

Karena a0 = 1, fungsi pembangkit terkait adalah:

g(x) = 1 + a1x + a2x2 + …

Dengan menempatkan n = 1, 2, 3, dalam anxn = 2an–1xn – (n/3)xn, diperoleh:

$$a_1x = 2x - \tfrac{1}{3}x$$
$$a_2x^2 = 2a_1x^2 - \tfrac{2}{3}x^2$$
$$\vdots$$

Jadi dengan menambahkan persamaan ini, kami memiliki:

$$g(x) - 1 = 2xg(x) - \frac{x}{3}(1 + 2x + 3x^2 + \cdots)$$

$$(1 - 2x)g(x) = 1 - \frac{x}{3}f(x)$$

di mana:

$$f(x) = 1 + (1 + 1)x + (1 + 1 + 1)x^2 + \cdots = \frac{1}{(1 - x)^2}$$

[Ingat bahwa jika u(x) adalah fungsi pembangkit biasa untuk pr, maka u(x)/(1 – x) adalah fungsi pembangkit biasa untuk qr = p1 + p2 + … + pr. Dalam kasus ini 1/(1 – x)2 adalah fungsi pembangkit untuk pr = 1. Jadi 1/(1 – x)2 adalah fungsi pembangkit untuk qr = 1 + 1 + … + 1 = r.] Jadi dari persamaan fungsional untuk g(x) kita dapatkan:

$$g(x) = \frac{3 - 7x + 3x^2}{3(1 - x)^2(1 - 2x)}$$

Sekarang fungsi di ruas kanan dapat diperluas dengan metode pecahan parsial. Kami kemudian memiliki:

$$g(x) = \left(\frac{1}{3}\right)\left[\frac{1}{1-x} + \frac{1}{(1-x)^2} + \frac{1}{1-2x}\right]$$

Dengan demikian:

$$\begin{aligned} a_n &= \text{coefficient of } x^n \text{ in } g(x) \\ &= \frac{1}{3}[1 + (n+1) + 2^n] = \frac{n+2+2^n}{3} \end{aligned}$$

Contoh 4.4.3

Temukan fungsi pembangkit untuk relasi perulangan an = c1an–1 + c2an–2 dengan a0 = A0 dan a1, = A1.

Bukti:

Fungsi pembangkit adalah:

g(x) = a0 + a1x + a2x2 + …

Menempatkan n = 2, 3, 4, . . . dalam anxn = c1an–1xn + c2an–2xn, kita memiliki:

$$\begin{aligned} a_2x^2 &= c_1a_1x^2 + c_2a_0x^2 \\ a_3x^3 &= c_1a_2x^3 + c_2a_1x^3 \\ &\vdots \end{aligned}$$

Selain itu, kami memiliki:

g(x) – A0 – A1x = (c1x)[g(x) – A0] + (c2x2)g(x)

Jadi g(x) = u(x)/v(x) , di mana u(x) = (A0 + A1x) – (A0)(c1x) dan v(x) =1 – c1x – c2x2.

Perhatikan hubungan antara koefisien penyebut v(x) dan koefisien fungsi karakteristik p(x) = x2 – c1x – c2. Kita dapat menggeneralisasi pengamatan ini untuk relasi rekurensi homogen linier berorde r dengan koefisien konstan sebagai berikut.

TEOREMA 4.4.1

Jika p(x) dan g(x) adalah fungsi karakteristik dan fungsi pembangkit dari relasi rekurensi homogen linier dengan koefisien konstan an = c1an_1 + c2an-2 + • • • + cran-r dengan kondisi awal berurutan ai, = Ai , (i = 0, 1, 2, . . , r – 1), maka p(x) = xr – c1xr–1 – c2xr–2 – • • • – cr dan g(x) = u(x) /v(x) , di mana penyebutnya adalah 1 – cxx – c2x2 – • • • –crxr dan pembilangnya adalah:

$$\begin{aligned} &[A_0 + A_1x + \cdots + A_{r-1}x^{r-1}] - (c_1x)[A_0 + A_1x + \cdots + A_{r-2}x^{r-2}] \\ &\quad - (c_2x^2)[A_0 + A_1x + \cdots + A_{r-3}x^{r-3}] - \cdots - (c_{r-1}x^{r-1})[A_0] \end{aligned}$$

Contoh 3.4.4
Jika $p(x) = x3 – 9x2 + 26x – 24$, cari $g(x)$.
Bukti:
Fungsi pembangkit $g(x)$ adalah $u(x)/v(x)$,
Dimana:
$v(x) = 1 – 9x + 26x2 – 24x3$
$u(x) = (A0 + A1x + A2x2) – (9x)(A0 + A1x) + (26x2)(A0)$

4.5 ANALISIS ALGORITMA

Pertama-tama kita membuat perbedaan antara "masalah" dan "contoh masalah". Misalnya, menemukan produk dari dua bilangan bulat adalah masalah, sedangkan menemukan produk dari dua bilangan bulat yang diberikan adalah contoh dari masalah. Jika masalahnya adalah menemukan produk dari dua matriks persegi, maka menemukan produk dari dua matriks $n \times n$ adalah turunan dari masalahnya. Dalam optimasi jaringan, masalah jarak terpendek adalah masalah menemukan jarak terpendek dari suatu simpul ke simpul lain dalam suatu jaringan. Sebuah contoh dari masalah ini kemudian akan menemukan jarak terpendek dari sebuah simpul ke simpul yang tersisa dalam jaringan yang diberikan dengan n simpul dan m tepi. Dalam pengertian informal dan intuitif, algoritma untuk masalah, seperti yang kita semua tahu, adalah prosedur langkah demi langkah yang melibatkan urutan instruksi terbatas yang dapat digunakan untuk menyelesaikan setiap contoh masalah.

Dari sudut pandang informal, kompleksitas komputasi dari suatu algoritma untuk suatu masalah adalah biaya, diukur dalam waktu berjalan atau penyimpanan atau unit apa pun yang relevan, dari penggunaan algoritma untuk memecahkan masalah. Biaya ini, dilambangkan dengan $f(n)$, tergantung pada ukuran input n dari sebuah instance masalah. Fungsi f adalah fungsi kompleksitas dari algoritma. Dalam analisis algoritme, kebutuhan biaya ini biasanya dinyatakan dalam jumlah langkah komputasi dasar, seperti operasi aritmatika, perbandingan, dan sebagainya, yang diperlukan untuk eksekusi algoritme pada komputer (hipotetis) dengan asumsi bahwa semua jenis operasi ini membutuhkan waktu satuan. Sangat mungkin bahwa persyaratan biaya untuk dua contoh berbeda dengan ukuran input yang sama dapat berbeda secara signifikan. Jadi kami mempertimbangkan semua contoh dari ukuran input yang diberikan n dan kemudian memilih persyaratan biaya dari contoh yang biayanya maksimum. Dengan kata lain, kita sedang memeriksa perilaku kasus terburuk dari algoritma. Jika A adalah algoritma untuk memecahkan masalah, kami menyatakan dengan $f(A, n)$ kompleksitas kasus terburuk dari algoritma, di mana n adalah ukuran input. Misalnya, pertimbangkan masalah memilih bilangan terkecil dari sekumpulan n bilangan. Jika kita tidak memiliki informasi tentang angka-angka ini dan jika kita memilih angka terkecil setelah membuat semua perbandingan yang mungkin, jumlah perbandingan adalah $(n – 1)$ dan oleh karena itu kompleksitas kasus terburuk untuk prosedur pemilihan ini adalah $(n – 1)$.

Sebagai contoh lain, pertimbangkan algoritma perkalian matriks biasa, yang kita semua kenal. Misalkan L dan M adalah sembarang dua matriks $n \times n$ sedemikian rupa sehingga tidak ada elemen dalam kedua matriks yang nol. Dengan demikian, kami memiliki skenario terburuk di sini. Dalam mencari matriks produk kita tidak perlu khawatir tentang penambahan. Sebagai gantinya, kami menghitung jumlah total perkalian yang terlibat dalam penerapan algoritma.

Perhatikan bahwa setiap elemen dalam matriks produk adalah produk dari baris r dari L dan kolom c dari M. Baik r dan c memiliki n bilangan bukan nol. Jadi, hasil kali melibatkan n perkalian. Karena ada n2 elemen dalam matriks produk, jumlah total perkalian adalah f(A, n) = n3 yang merupakan kompleksitas kasus terburuk untuk perkalian matriks, di mana A adalah algoritma perkalian matriks biasa.

Setelah algoritma ditentukan untuk suatu masalah, kita akan menulis f(n) alih-alih f(A, n) untuk menunjukkan kompleksitas kasus terburuk dari algoritma. Di sisi lain, jika kita membandingkan dua algoritma A dan A′ untuk masalah yang sama, kita menulis f(A, n) dan f(A′, n) untuk menunjukkan kompleksitas masing-masing dan dalam hal ini kita katakan bahwa A adalah lebih efisien daripada A jika f(A, n) tidak melebihi f(A , n) untuk semua n ketika n n0 untuk beberapa bilangan bulat positif tetap n0. Jika f(A, n) = 5n2 dan f(A , n) = n3 maka A lebih efisien daripada A karena f(A, n) ≤ f(A , n) untuk semua n 5.

Dalam banyak kasus, dimungkinkan untuk membagi masalah menjadi beberapa submasalah kecil yang tidak tumpang tindih dengan ukuran yang kira-kira sama, memecahkan submasalah ini, dan menggabungkan solusi mereka untuk mendapatkan solusi dari masalah asli. Strategi pemecahan masalah ini, yang dikenal sebagai teknik membagi-dan-menaklukkan, seringkali lebih efisien daripada metode langsung biasa. Jika f(A, n) adalah kompleksitas komputasi dari algoritma membagi-dan-menaklukkan (di mana n adalah ukuran input dari sebuah instance dari masalah) kita memiliki hubungan pengulangan yang menyatakan f(A, n) dalam bentuk f (A, m) , di mana m adalah ukuran input dari turunan dari submasalah. Pada penyelesaian hubungan perulangan ini menggunakan kondisi awal kita memperoleh fungsi kompleksitas dari algoritma. Kami sekarang mengalihkan perhatian kami ke analisis beberapa hubungan perulangan ini yang dihasilkan dari algoritma membagi-dan-menaklukkan tersebut.

Contoh 4.5.1

Jika a dan b adalah dua bilangan n-digit untuk mendapatkan hasil kali ab, kita harus melakukan paling banyak n2 perkalian satu digit. Dengan kata lain, jika kita menggunakan algoritma perkalian biasa, kompleksitas komputasinya adalah n2 jika kita mengabaikan jumlah penjumlahan dan hanya memperhitungkan jumlah perkalian total. Kita dapat menggunakan algoritma bagi-dan-taklukkan yang lebih efisien untuk mengalikan dua angka sebagai berikut. Asumsikan bahwa n genap. Kemudian baik a dan b dapat dibagi menjadi dua bagian:

$$a = a1(10)n/2 + a2 \quad b = b1(10)n/2 + b2$$

di mana bagian a1, a2, b1, dan b2 semuanya (n/2)-angka angka.
Kemudian:

$$ab = a1b1(10)n + [a1b2 + a2b1](10)n/2 + a2b2$$

Ekspresi di sebelah kanan melibatkan perkalian empat (n/2)-digit. Kita dapat mengurangi ini menjadi tiga perkalian jika kita menggunakan rumus:

$$(a1b2 + a2b1) = (a1 + a2)(b1 + b2) – a1b1 – a2b2$$

Jadi kita memiliki relasi perulangan f(n) = 3f(n/2) dengan f(l) = 1. Tentu saja, ada kemungkinan bahwa (a1 + a2) atau (b1+ b2) mungkin (n/2 + 1)-digit angka, tetapi ini tidak mempengaruhi pertimbangan kompleksitas. Jika kita menulis n = 2k dalam hubungan perulangan, kita mendapatkan:

$$f(n) = 3f\left(\frac{n}{2}\right) = 3^2 f\left(\frac{n}{2^2}\right) = \cdots = 3^k f(1) = 3^{\log n} = n^{\log 3}$$

Jadi, kompleksitas algoritma rekursif ini adalah nlog3, yang lebih kecil dari n2 untuk semua n 1.

Secara lebih umum, mari kita pertimbangkan algoritma bagi-dan-taklukkan yang membagi masalah berukuran n menjadi beberapa submasalah yang masing-masing berukuran n/b. Kita asumsikan bahwa n = bk dan jumlah submasalah adalah a, dimana a > 1. Jadi jika kompleksitas algoritma untuk masalah adalah f(n), maka kompleksitas algoritma yang sama untuk submasalah adalah f(t/b) . Dengan demikian, kita memiliki relasi perulangan dalam bentuk f(n) = af(n/b) + h(n), f(1) = d, a > 1, n = bk , di mana h(n) mewakili biaya membagi masalah menjadi submasalah ditambah biaya penggabungan solusi dari submasalah tersebut untuk mendapatkan solusi dari masalah aslinya. Dua kasus khusus yang sangat penting: (1) h(n) adalah konstanta c dan (2) h(n) = cn, di mana c adalah konstanta.

Contoh 4.5.2

Selesaikan f(n) = af(n/b) + c, f(l) = d, di mana a dan b adalah bilangan bulat yang paling sedikit 2, n = bk , dan c adalah konstanta.

Bukti:

$$\begin{aligned} f(n) &= af\left(\frac{n}{b}\right) + c = a\left[af\left(\frac{n}{b^2}\right) + c\right] + c \\ &= a^2 f\left(\frac{n}{b^2}\right) + c[1 + a] \\ &\vdots \\ &= a^k f\left(\frac{n}{b^k}\right) + c[1 + a + a^2 + \cdots + a^{k-1}] \\ &= a^k d + c\frac{(a^k - 1)}{a - 1} \\ &= Aa^k + B \end{aligned}$$

di mana A = (iklan – d + c)/(a – 1) dan B = (–c)/(a – 1). Karena n = bk , kita memiliki k = logbn dan ak = nr, di mana r = logba. Jadi f(n) = Anr + B.

Contoh 4.5.3

Selesaikan f(n) = af(n/b) + cn dengan kondisi yang sama seperti pada Contoh 3.5.2.

Bukti:

Dengan iterasi kita memiliki

$$f(n) = a^k f(1) + cn\left[1 + \frac{a}{b} + \left(\frac{a}{b}\right)^2 + \cdots + \left(\frac{a}{b}\right)^{k-1}\right]$$

Kasus (1): a = b

f(n) = akd + cnk = bkd + cnk = nd + cn logbn

Jadi f(n) = c(n log n) + d(n) .

Kasus (2): a < b atau a > b. Setelah disederhanakan, kita mendapatkan f(n) = Anr + Bn, di mana A = (bd – ad – bc)/(b – a) dan B = bc/(b – a) dan r = logba. Jika a < b, maka r < 1, maka Bn adalah suku yang mendominasi f(n) . Jika a > b, maka r > 1, maka Anr adalah suku yang mendominasi f(n).

Perkalian Matriks

Kami selanjutnya mempertimbangkan efisiensi algoritma perkalian matriks. Seperti yang kita ketahui sebelumnya, jika A dan B adalah dua matriks 2 × 2, kita biasanya melakukan setidaknya delapan (23 = 8) perkalian untuk mendapatkan matriks produk C = AB. Dengan manipulasi aljabar yang cerdas, dimungkinkan untuk mengurangi jumlah perkalian dari delapan menjadi tujuh. Misalkan a(i, j) menyatakan elemen dalam A pada baris ke-i dan kolom ke-j; sama untuk B dan C.

Sekarang tentukan tujuh produk berikut:

x1 = [a(1, 1) + a(2, 2)] • [b(1, 1) + b(2, 2)]
x2 = [a(2, 1) + a(2, 2)] • b(1, 1)
x3 = a(1, 1) • [b(1, 2) – b(2, 2)]
x4 = a(2, 2) • [b(2, 1) – a(2, 1)]
x5 = [a(1, 1) + a(1, 2)] • b(2, 2)
x6 = [a(2, 1) – a(1, 1)] • [b(1, 1) + b(1, 2)]
x7 = [a(1, 2) – a(2, 2)] • [b(2, 1) + b(2, 2)]

Maka mudah untuk memverifikasi bahwa setiap elemen c(i,j) dari matriks produk dapat dinyatakan sebagai jumlah atau selisih dari beberapa dari tujuh angka ini sebagai berikut:

c(1, 1) = x1 + x4 – x5 + x7 c(1, 2) = x3 + x5
c(2, 1) = x2 + x4 c(2, 2) = x1 + x3 – x2 + x6

Jadi kita mendapatkan matriks produk dengan melakukan paling banyak tujuh perkalian, bukan delapan. Kami menggunakan informasi ini untuk mendefinisikan algoritma bagi-dan-taklukkan untuk perkalian matriks dan memecahkan hubungan perulangan terkait untuk mendapatkan kompleksitas komputasi dari algoritma ini dalam contoh berikutnya.

Contoh 4.5.4

Misalkan A dan B adalah dua matriks n × n dimana n = 2k . Dapatkan algoritma bagi-dan-taklukkan untuk mendapatkan matriks produk AB dan temukan kompleksitas dari algoritma ini.

Bukti:

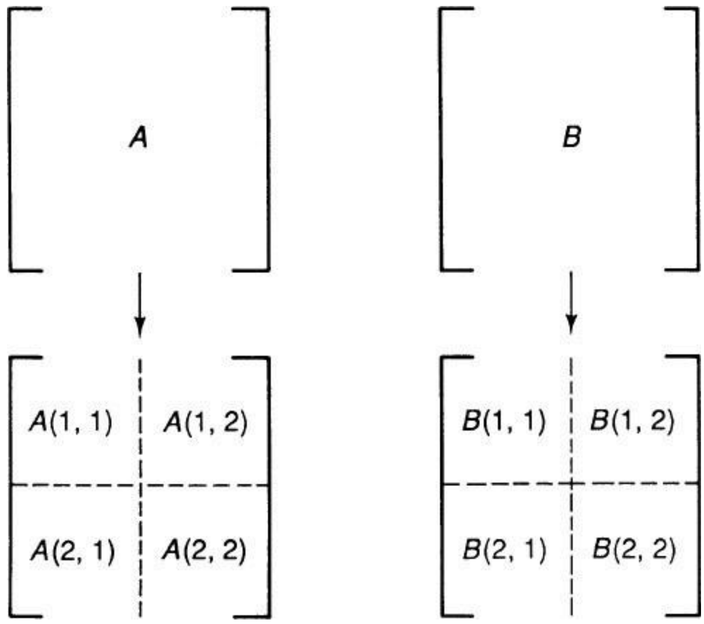
Kami mempartisi setiap matriks menjadi empat submatriks di mana setiap submatriks adalah matriks (n/2) × (n/2). Lihat Gambar 3.5.1.

Empat submatriks dari A adalah: (1) A(l, 1) diperoleh dengan mempertimbangkan n/2 baris pertama dan n/2 kolom pertama A; (2) A(l, 2) diperoleh dengan mempertimbangkan n/2 baris pertama dan n/2 kolom terakhir; (3) A(2, 1) diperoleh dengan mempertimbangkan n/2 baris terakhir dan n/2 kolom pertama; dan (4) A(2, 2) diperoleh dengan mempertimbangkan n/2 baris terakhir dan n/2 kolom terakhir dari A; sama untuk B dan C.

Kami sekarang mendefinisikan tujuh submatriks berikut:

X1 = [A(1, 1) + A(2, 2)] • [B(1, 1) + B(2, 2)]
X2 = [A(2, 1) + A(2, 2)] • B(1, 1)
X3 = A(1, 1) • [B(1, 2) – B(2, 2)]
X4 = A(2, 2) • [B(2, 1) – B(1, 1)]
X5 = [A(1, 1) + A(1, 2)] • B(2, 2)
X6 = [A(2, 1) – A(1, 1)] • [B(1, 1) + B(1, 2)]
X7 = [A(1, 2) – A(2, 2)] • [B(2, 1) + B(2, 2)]

Perhatikan bahwa masing-masing submatriks (n/2) × (n/2) ini adalah perkalian dari dua (n/2) × (n/2) submatriks. Dapat dengan mudah diverifikasi bahwa:



Gambar 3.1 Submatrik matriks A dan B

C(1, 1) = X1 + X4 – X5 + X7 C(1, 2) = X3 + X5
C(2, 1) = X2 + X4
C(2, 2) = X1 + X3 – X2 + X6

Jadi jika f(n) adalah jumlah perkalian yang diperlukan untuk mengalikan A dan B menggunakan strategi bagi-dan-taklukkan ini, kita memiliki hubungan perulangan f(n) = 7f(n/2) dengan kondisi awal f(1) = 1. Pada penyelesaian ini kita mendapatkan f(n) = 7k f(1) = 7k , di mana k = log2n. Secara ekivalen, f(n) = nr, di mana r = log27. Dengan demikian kompleksitas algoritma ini kurang dari n3, sehingga metode ini lebih efisien daripada metode perkalian matriks biasa!

Evaluasi Polinomial pada Titik yang Diberikan

Kami menyimpulkan analisis algoritma kami dengan diskusi tentang jumlah perkalian dan penambahan yang terlibat untuk mengevaluasi nilai polinomial p(x) untuk nilai x yang diberikan. Jika derajat p(x) adalah n, mudah untuk melihat bahwa evaluasi tak rekursif langsung dari p(x) pada x = t akan melibatkan (2n – 1) perkalian dan n penjumlahan. Misalnya, untuk mengevaluasi p(x) = 5x3 + 8x2 + 4x + 7 pada x = 27, kita melakukan lima perkalian (1) a = 27 • 27, (2) b = a • 27, (3) c = 4 • 27, (4) 8 • a, dan (5) 5 • b dan kemudian tiga penambahan. Metode yang lebih efisien (rekursif) yang dikenal sebagai metode Horner (atau metode Newton) hanya membutuhkan n perkalian dan n penambahan. Jika kita menulis polinomial dalam bentuk telescoping sebagai p(x) = (((5 • x + 8) • x) + 4) • x + 7, jumlah perkaliannya hanya 3.

Contoh 3.5.5 (Metode Horner)

(a) Jika f(n) adalah jumlah perkalian yang diperlukan untuk mengevaluasi polinomial berderajat n pada suatu titik, dapatkan relasi rekursif untuk f(n).
(b) Jika g(n) adalah jumlah perkalian dan penjumlahan, tentukan relasi rekursif yang melibatkan g(n) .

Bukti:

Jika p(x) adalah polinomial berderajat n, maka p(x) = xq(x) + a, di mana a adalah konstanta dari q(x) adalah polinomial berderajat (n – 1).

(a) Evaluasi q(x) pada suatu titik akan melibatkan perkalian f(n – 1). Jadi f(n) = f(n – 1) + 1, dengan kondisi awal f(0) = 0. Jadi f(n) = n.
(b) Jelas, g(n) = g(n – 1) + 2 dengan g(0) = 0. Solusinya adalah g(n) = 2n.

Akhirnya, kami membahas algoritma rekursif lain untuk evaluasi polinomial, yang lebih efisien daripada metode Horner. [Metode ini dijelaskan secara rinci dalam Baase (1978).] Pertama definisi dan beberapa sifat polinomial. Suatu polinomial p(x) disebut polinomial monik jika koefisien suku utamanya adalah 1. Kita juga mengasumsikan bahwa derajat n dari p(x) sama dengan 2k – 1. Banyaknya perkalian yang diperlukan untuk evaluasi polinomial untuk p (x) adalah (n – 1) = 2k – 2 menurut metode Horner.

Bisakah kita melakukan yang lebih baik mengenai jumlah perkalian? Sekarang dapat ditunjukkan bahwa jika p(x) = xn + an–1xn–1 + •• + a1x + a0 adalah polinomial monik berderajat 2k – 1, kita dapat menulis p(x) sebagai p(x) = (xj + b) • q(x) + r(x) di mana (1) q(x) dan r(x) keduanya adalah polinomial monik berderajat (n – 1)/2 dan j = (n + 1)/ 2, (2) b = aj–1 – 1, dan (3) koefisien q(x) adalah koefisien pertama (n + 1)/2 dari p(x) , dimulai dari suku tertinggi. Jika kekuatan x hilang, koefisien yang sesuai diambil sebagai nol. Misalnya, mari:

p(x) = x7 + 2x6 + 2x5 + 3x4 + 9x3 + 9x2 + 18x + 9

Di sini 2k – 1 = 7. Jadi k = 3 dan j = 4. Juga, b = a3 – 1 = 9 – 1 = 8. Jadi p(x) = (x4 + 8)q(x) + r(x) , di mana q(x) adalah polinomial berderajat 3 yang koefisiennya 1, 2, 2, 3 dimulai dari suku tertinggi. Dengan kata lain, q(x) = x3 + 2x2 + 2x + 3 dan r(x) = p(x) – (x4 + 8)q(x) = x3 – 7x2 + 2x – 15
Melanjutkan dengan cara yang sama kita menulis:

$$q(x) = (x2 + 1)(x + 2) + (x + 1)$$

dan

$$r(x) = (x2 + 1)(x – 7) + (x – 8)$$

Jadi akhirnya, kita punya:

$$p(x) = (x4 + 8) \bullet [(x2 + 1) \bullet (x + 2) + (x + 1)] + (x2 + 1) \bullet (x – 7) + (x – 8)$$

yang melibatkan lima perkalian semuanya (tiga seperti yang ditunjukkan dengan tanda kurung dan dua untuk menghitung x2 dan x4). Di sisi lain, jika kita menggunakan metode Horner, jumlah perkalian akan menjadi enam, bukan lima. Pengamatan ini digeneralisasikan sebagai berikut.

TEOREMA 4.5.1

Misalkan p(x) adalah polinomial monik berderajat n, di mana n = 2k – 1. Maka banyaknya perkalian yang diperlukan untuk mengevaluasi p(x) pada suatu titik paling banyak (n – 3)/2 + log(n + 1), dan jumlah penambahan/pengurangan yang dibutuhkan paling banyak (3n – 1)/2.

Bukti:

Misal n = 2k – 1. Juga misalkan f(k) adalah jumlah perkalian yang diperlukan untuk mengevaluasi p(x) (tanpa memperhitungkan jumlah perkalian yang diperlukan untuk menghitung berbagai pangkat x pada titik yang diberikan), dan biarkan g(k) adalah jumlah tambahan yang dibutuhkan. Karena p(x) = (xj + b)q(x) + r(x), kita memiliki relasi perulangan f(k) = 2f(k – 1) + 1 dengan kondisi awal f(1) = 0 dan relasi g(k) = 2g(k – 1) + 2 dengan g(1) = 1. Pada penyelesaian hubungan perulangan ini, kita memiliki:

$$f(k) = 2^{k-1} - 1 = \frac{n-1}{2} \qquad \text{and} \qquad g(k) = 3 \cdot 2^{k-1} - 2 = \frac{3n-1}{2}$$

Selanjutnya kita mempertimbangkan jumlah perkalian yang terlibat untuk menghitung berbagai kekuatan x. Kita harus menghitung x2, lalu x4, lalu x8, dan seterusnya, sampai kita mencapai pangkat ke-j dari x, di mana j = (n + 1)/2 = 2k -1, dan proses ini akan melibatkan (k – 1) perkalian. Tapi k = log(n + 1). Jadi algoritma ini membutuhkan paling banyak (n – 1)/2 + log(n + 1) – 1 = (n/2) + log(n + 1) – (3/2) perkalian dan (3n – 1)/ 2 penambahan, sedangkan metode Horner paling banyak membutuhkan (n – 1) perkalian dan n penambahan.

4.6 CATATAN DAN REFERENSI

Karya perintis dalam studi hubungan perulangan dilakukan oleh Leonardo dari Pisa (lebih dikenal sebagai Fibonacci) pada abad ketiga belas dan kemudian oleh Jacob Bernoulli (1654-1705), keponakannya Daniel Bernoulli (1692-1770), James Stirling (1692-1770), dan Leonhard Euler (1707-1783). Beberapa referensi yang berguna adalah bab-bab yang relevan dalam buku-buku tentang kombinatorik dan matematika diskrit oleh Cohen (1978), Grimaldi (1985), Krishnamurthy (1986), Liu (1968), Liu (1985), Roberts (1984), Townsend (1987) , dan Tucker (1984). Teknik untuk memecahkan hubungan rekurensi pertama kali digunakan dalam pengembangan teori persamaan perbedaan. Lihat Levy dan Lessman (1961) untuk survei metode ini. Sebuah referensi yang sangat baik untuk aplikasi persamaan perbedaan adalah Goldberg (1958). Untuk bacaan tambahan tentang analisis algoritma, lihat bab-bab yang relevan dalam buku-buku oleh Baase (1978), Knuth (1973), Roberts (1984), Stanat dan McAllister (1977), dan Wilf (1986).

4.7 LATIHAN

4.1 Misalkan terdapat n garis pada suatu bidang sedemikian rupa sehingga tidak ada dua garis yang sejajar dan tidak ada tiga garis yang konkuren membagi bidang tersebut menjadi f(n) daerah yang berbeda. Temukan relasi perulangan untuk f(n) dan selesaikan f(n). Temukan nilai f(9).

4.2 Carilah relasi perulangan untuk f(n), banyaknya n huruf kata yang dapat dibentuk dengan menggunakan huruf A, B, C, D, dan E sedemikian sehingga frekuensi setiap kata dari huruf A adalah bilangan ganjil.

4.3 (Masalah Menara Hanoi.) Tiga kutub silinder vertikal dengan jari-jari dan tinggi yang sama ditempatkan di sepanjang garis di atas meja dan n piringan bundar dengan jari-jari menurun, masing-masing dengan lubang di tengahnya, dipasang pada kutub pertama sedemikian rupa. bahwa yang terbesar ada di bawah, yang terbesar berikutnya tepat di atas itu, dan seterusnya, dan akhirnya yang terkecil ada di atas tumpukan (Gambar 3.7.1). Jarak antara kaki dua kutub tidak kurang dari diameter piringan terbesar. Sebuah langkah hukum didefinisikan sebagai transfer disk atas dari salah satu dari tiga kutub ke kutub lain sedemikian rupa sehingga tidak ada disk yang ditempatkan di atas disk yang lebih kecil. Misalkan f(n) adalah jumlah gerakan legal yang diperlukan untuk memindahkan semua piringan dari kutub pertama ke salah satu dari dua kutub lainnya. Dapatkan relasi perulangan untuk f(n) dan selesaikan relasi ini.

4.4 Dalam menaiki tangga, anak tangga biasa meliputi paling sedikit satu anak tangga dan paling banyak dua anak tangga. Jika f(n) adalah banyaknya cara menaiki sebuah tangga (hanya membuat anak tangga biasa) dengan n anak tangga, tentukan relasi perulangan untuk f(n) .

4.5 Misalkan S adalah himpunan semua kata biner dengan panjang n sedemikian rupa sehingga dua nol tidak muncul secara berurutan pada kata apa pun dalam himpunan tersebut. Temukan relasi perulangan untuk jumlah elemen di S.

4.6 Ada n cek pribadi yang dibuat atas nama n orang yang namanya tertera pada n amplop. Temukan relasi perulangan untuk banyaknya cara cek ini dapat ditempatkan dalam n amplop sehingga tidak ada cek dalam amplop yang benar.

4.7 Tunjukkan bahwa relasi perulangan $f(n) = (n - 1)[f(n - 1) + f(n - 2)]$ dengan $f(1) = 0$ dan $f(2) = 1$ dapat disederhanakan menjadi $f(n) = nf(n - 1) + (-1)n$ dengan $f(2) = 1$.

4.8 Misalkan f(n) adalah banyaknya himpunan bagian dari suatu himpunan yang memiliki n anggota. Temukan relasi perulangan untuk f(n) .

4.9 Misalkan f(n) adalah banyaknya anggota X yang merupakan himpunan semua kata bersimbol n yang dibentuk oleh simbol A, B, dan C sedemikian rupa sehingga tidak ada kata yang memiliki pasangan A berurutan.

4.10 Selesaikan untuk k dalam relasi perulangan $f(n + 1) = kf(n)$ jika (a) $f(1) = 5$ dan $f(2) = 10$, dan (b) $f(1) = 5$ dan $f(3) = 20$.

4.11 Selesaikan: $f(n + 3) = 6f(n + 2) - 11f(n + 1) + 6f(n)$, di mana $f(0) = 3$, $f(1) = 6$, dan $f(2) = 14$.

4.12 Selesaikan: $f(n + 3) = 4f(n + 2) - 5f(n + 1) + 2f(n)$, di mana $f(0) = 2$, $f(1) = 4$, dan $f(2) = 7$.

4.13 Selesaikan: $f(n + 3) = 3f(n + 2) + 4f(n + 1) - 12f(n)$ di mana $f(0) = 0$, $f(1) = -11$, dan $f(2) = -15$.

4.14 Akar persamaan karakteristik dari relasi rekurensi homogen linier dengan koefisien konstan adalah 1, 2, 2, dan 3. Tuliskan relasi tersebut dan solusi umumnya.

4.15 Selesaikan: $nf(n) - (5n - 5) f(n - 1) = 0$ di mana $f(1) = 10$. [Petunjuk: Pengganti $g(n) = nf(n)$.]

4.16 Misalkan A adalah matriks $m \times m$ yang semua bilangan diagonalnya sama dengan 0 dan semua bilangan nondiagonalnya sama dengan 1. Maka bilangan diagonal dari An semuanya sama dengan bilangan bulat positif f(n) dan bilangan nondiagonal dari Semua sama dengan bilangan bulat positif g(n) untuk sembarang bilangan bulat positif n. Buktikan bahwa $f(n + 1) = (m - 1)g(n)$ dan $g(n + 1) = f(n) + (m - 2)g(n)$ dan gunakan fakta ini untuk memperoleh relasi perulangan untuk g(n) dengan kondisi awal yang sesuai. Memecahkan hubungan. Temukan g(n) dan f(n).

4.17 Selesaikan relasi perulangan tak homogen berikut yang melibatkan f(n): $f(n) - 4f(n - 1) + 4f(n - 2) = h(n)$ di mana (a) $h(n) = 1$, (b) $h(n) = n$, (c) $h(n) = 3n$, (d) $h(n) = 2n$, dan (e) $h(n) = 1 + n + 2n + 3n$.

4.18 Selesaikan relasi $f(n + 2) - 4/(n + 1) + 3f(n) = 16$ dengan kondisi awal $f(0) = 4$ dan $f(1) = 2$.

4.19 Selesaikan: $f(n) = 4f(n - 1) + 5(3)n$.

4.20 Selesaikan: $f(n) = 4f(n - 1) + 5(4)n$.

4.21 Selesaikan: $f(n) = f(n - 1) + 2f(n - 2) + 4(3)n$ dengan kondisi awal $f(0) = 11$ dan $f(1) = 28$.

4.22 Selesaikan: $f(n) = 4f(n - 1) - 4f(n - 2) + (2)n$.

4.23 Selesaikan relasi perulangan $f(n) = f(n - 1) + 6n2$, $f(0) = 0$, dengan (a) menggunakan akar karakteristik, dan (b) substitusi berulang. Jadi, tentukan jumlah kuadrat dari n bilangan asli pertama.

4.24 Tentukan konstanta p, q, dan r dalam relasi perulangan $f(n) + pf(n - 1) + qf(n - 2) = r$ jika $f(n) = A(2)n + B(3)n + 4$.

4.25 Jika fungsi pembangkitan biasa dari relasi perulangan yang melibatkan f(n) adalah $g(x) = (2)/[(1 - x) (1 - 2x)]$, cari f(n).

4.26 Selesaikan relasi perulangan $f(n) = f(n - 2) + 4n$, $f(1) = 2$, $f(0) = 3$, dengan menggunakan fungsi pembangkit biasa yang sesuai.

4.27 Buktikan bahwa jika g(x) adalah fungsi pembangkit biasa untuk relasi perulangan f(n + 1) = (n + 1)f(n) + (–1)n+1 dengan kondisi awal f(0) = 1 , maka g(x) memenuhi persamaan diferensial g′(x) + [(x – 1)/x2]g{x) + (1)/(x2) (1 + x) = 0. Selesaikan relasi perulangan dengan menggunakan fungsi pembangkit eksponensialnya.

4.28 Relasi perulangan untuk algoritma divide-and-conquer adalah f(n) = 9f(n/3) + 8n dengan kondisi awal f(1) = 1, dimana n = 3r. Selesaikan f(n) sebagai fungsi dari n.

4.29 Selesaikan relasi perulangan f(n) = 5f(n/2) – 6f(n/4) + n dengan kondisi awal f(1) = 2 dan f(2) = 1, dimana n = 2r.

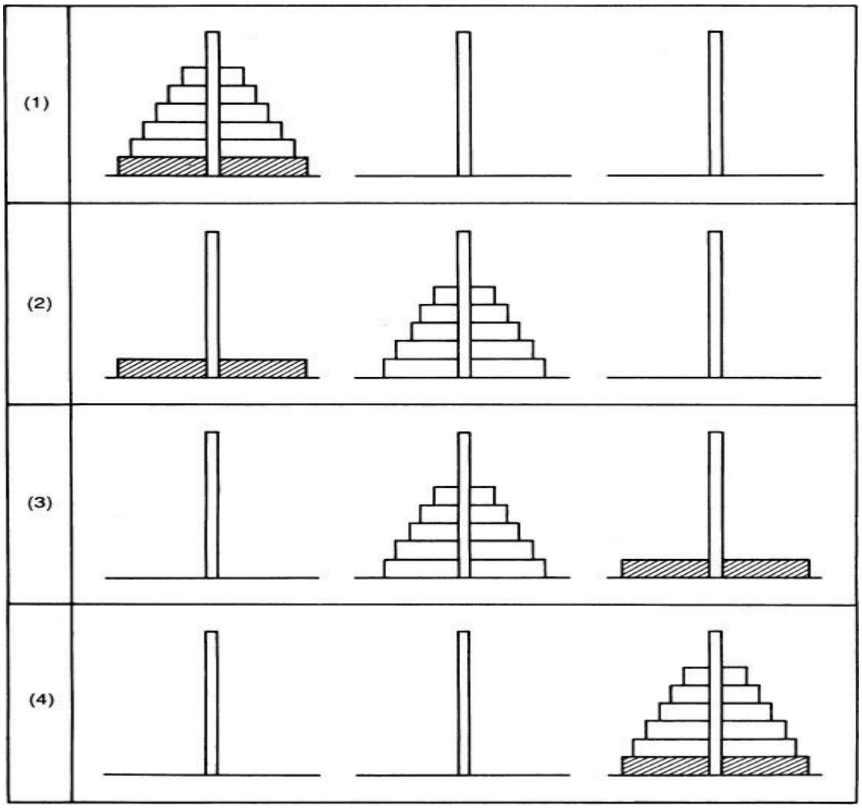
4.30 Selesaikan f(n) = f(n/b) + c dengan kondisi awal f(1) = d, di mana n = br.

4.31 Tentukan fungsi pembangkit biasa untuk relasi perulangan f(n + 1) = af(n) + bn dengan kondisi awal f(0) = c, di mana a, b, dan c adalah konstanta.

4.32 Perhatikan contoh pada bagian ini yang membahas kompleksitas perkalian matriks. Jika f(n) adalah jumlah perkalian yang diperlukan untuk mencari hasil kali dua matriks n × n, terbukti bahwa f(n) = nr, di mana r = log 7. Carilah relasi rekursif untuk banyaknya penjumlahan yang terlibat dan menyelesaikannya.

4.33 Temukan relasi perulangan (melibatkan jumlah perbandingan) untuk algoritma bagi-dan-taklukkan untuk menemukan elemen terbesar dan terkecil dalam himpunan n angka dan menyelesaikannya.

4.34 Misalkan f(n) adalah jumlah perbandingan yang diperlukan untuk mengurutkan daftar n angka dalam urutan tak menurun. (a) Dapatkan relasi rekursif yang menyatakan f(n) dalam suku f(n – 1) dengan kondisi awal yang sesuai. (b) Dapatkan relasi rekursif yang menyatakan f(n) dalam bentuk f(n/2) dengan kondisi awal yang sesuai, (c) Selesaikan kedua relasi perulangan ini dan bandingkan efisiensi kedua algoritma yang terlibat.



Gambar 4.2 Relasi rekursif

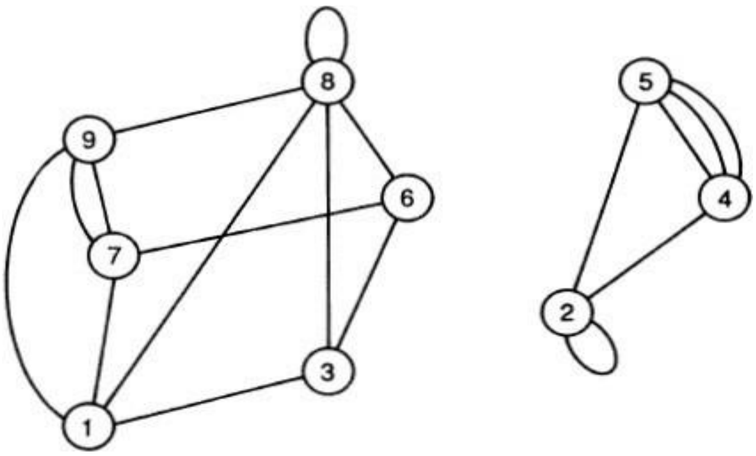
BAB 5
GRAFIK DAN DIGRAF

5.1 PENDAHULUAN

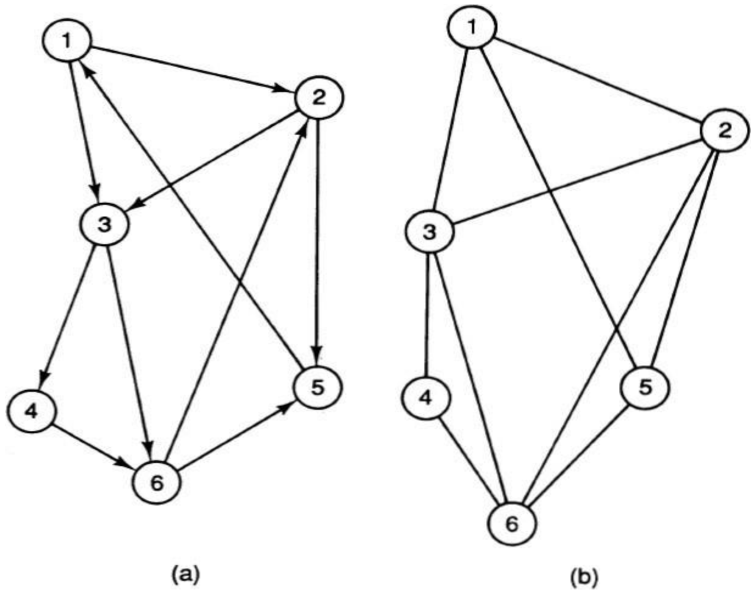
Meskipun asal-usul teori graf dapat ditelusuri kembali ke zaman matematikawan besar Swiss Leonhard Euler (1707-1783), hanya sejak tahun 1930-an telah ada minat yang berkelanjutan dan intens dalam teori graf sebagai disiplin matematika. Saat ini, teori graf adalah salah satu cabang yang paling populer dan subur dalam matematika dan ilmu komputer. Salah satu alasan penting untuk minat yang dihidupkan kembali dan diperbarui ini dalam teori graf adalah penerapannya pada banyak masalah masyarakat modern yang kompleks dan luas di berbagai bidang seperti ekonomi, lokasi fasilitas, ilmu manajemen, pemasaran, pemodelan energi, transmisi informasi, dan perencanaan transportasi untuk beberapa nama. Cukup sering masalah seperti itu dapat dimodelkan sebagai grafik atau jaringan. Dalam hal ini teori graf digunakan pertama dan terutama sebagai alat untuk merumuskan masalah dan mendefinisikan keterkaitan struktural. Setelah masalah dirumuskan dalam bahasa teoretis grafik, menjadi relatif mudah untuk memahaminya secara umum. Langkah selanjutnya, tentu saja, adalah mencari jalan untuk mencari solusi atas masalah tersebut. Bidang teori graf memiliki dua cabang yang berbeda: aspek aljabar dan aspek optimasi. Dalam Bab 4, 5, dan 6 kita membahas yang pertama. Bidang optimasi jaringan, yang sangat maju dengan munculnya komputer, adalah topik untuk dua bab terakhir.

Graf G = (V, E) adalah struktur yang terdiri dari himpunan berhingga V dari simpul (juga dikenal sebagai simpul) dan himpunan berhingga E tepi sedemikian rupa sehingga setiap tepi e dikaitkan dengan sepasang simpul v dan w. Kami menulis e = {v, w} atau {w, v} dan mengatakan bahwa (1) e adalah rusuk antara v dan w, (2) e bersinggungan dengan v dan w, dan (3) e menghubungkan simpul v dan w. Dalam hal ini baik v dan w adalah simpul-simpul yang bertetangga dan keduanya bersisian pada e. Sebuah edge yang menghubungkan sebuah vertex dengan dirinya sendiri adalah sebuah loop. Jika ada lebih dari satu sisi yang menghubungkan pasangan simpul dalam suatu graf, maka graf tersebut merupakan graf multigraf. Jika dua atau lebih sisi bergabung dengan pasangan simpul yang sama dalam sebuah multigraf, sisi-sisi ini disebut sisi ganda. Dalam representasi bergambar dari grafik, sebuah simpul digambar sebagai lingkaran kecil dengan nama (atau nomor) dari simpul yang ditulis di dalam lingkaran. Tepi antara dua simpul diwakili oleh segmen garis atau kurva yang menghubungkan dua lingkaran yang mewakili simpul. Pada Gambar 5.1 kita memiliki representasi bergambar dari sebuah multigraf di mana himpunan simpulnya adalah V = {1, 2, 3, 4, 5, 6, 7, 8, 9} dengan loop di simpul 2 dan di simpul 8. Ada dua sisi antara 7 dan 9 dan tiga sisi antara 4 dan 5. Graf dikatakan sederhana jika tidak memiliki loop atau sisi ganda. Jika suatu bilangan real dikaitkan dengan setiap sisi, maka G adalah jaringan atau graf berbobot.

Graf atau digraf berarah adalah struktur G = (V, E) di mana lagi V adalah himpunan berhingga dari simpul dan E adalah himpunan berhingga busur sedemikian rupa sehingga setiap busur e di E dikaitkan dengan pasangan berurut dari simpul v dan w. Kita tulis e = (v, w) dan kita katakan bahwa (1) e busur dari v ke w, (2) simpul v bertetangga dengan simpul w, (3) simpul w bertetangga dengan simpul v, (4) busur e datang dari v, dan (5) busur e datang ke w. Dua simpul bertetangga jika ada busur dari satu simpul ke simpul lainnya. Kami memiliki digraf berbobot atau jaringan terarah setiap kali bilangan real dikaitkan dengan setiap busur. Jika kita memperlakukan setiap busur digraf sebagai sisi, struktur yang dihasilkan disebut graf yang mendasari digraf tersebut. Dalam representasi bergambar digraf, busur dari titik v ke titik w digambar sebagai segmen berarah dengan kepala panah mengarah ke w. Pada Gambar 5.2(a) kita memiliki representasi bergambar dari sebuah digraf yang grafik dasarnya adalah seperti pada Gambar 5.2(b). Dalam graf campuran G = (V, E) setidaknya satu elemen dari E adalah busur dan setidaknya satu elemen adalah sisi. Elemen dalam kategori sebelumnya adalah busur berarah, sedangkan elemen dalam kategori terakhir adalah tepi tidak berarah. Bilangan real yang terkait dengan tepi dan busur dalam jaringan biasanya ditulis di sepanjang tepi dan busur ini.

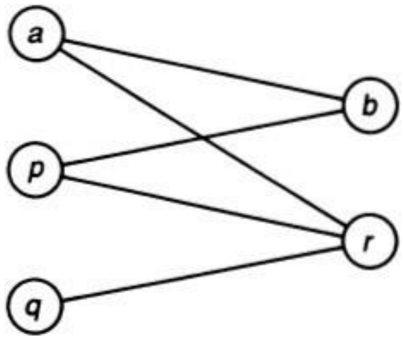


Gambar 5.1 Representasi multifgraf



Gambar 5.2 Representasi bergambar digraf a dan b

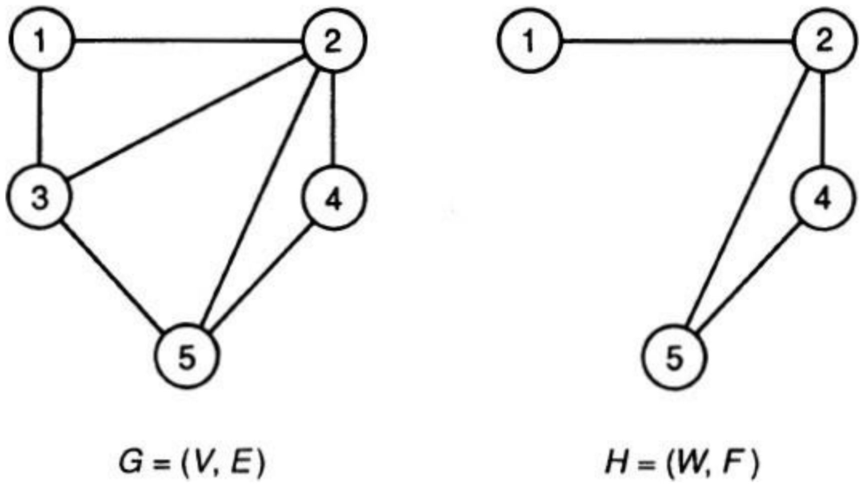
Yang menarik adalah graf bipartit yang simpul-simpulnya dapat dipartisi menjadi dua himpunan lepas V dan W sedemikian rupa sehingga setiap sisi adalah sisi antara sebuah simpul di V dan sebuah simpul di W dan dilambangkan dengan G = (V, W; E). Pada Gambar 5.3 kita memiliki representasi piktorial dari graf bipartit di mana V = {a, p, q} dan W = {b, r}. G = (V, W; E) adalah digraf bipartit jika setiap busur di E berasal dari titik di V ke titik di W.



Gambar 5.3 Representasi Piktorial

Graf sederhana dengan n simpul dikatakan lengkap jika terdapat rusuk di antara setiap pasangan simpul. Grafik tersebut kemudian dilambangkan dengan Kn. Digraf adalah digraf lengkap jika graf dasarnya lengkap. Graf bipartit sederhana G = (V, W, E) merupakan graf bipartit lengkap jika terdapat rusuk antara setiap simpul di V dan setiap simpul di W. Graf bipartit dilambangkan dengan Kp,q jika terdapat p simpul di V dan q simpul di W.

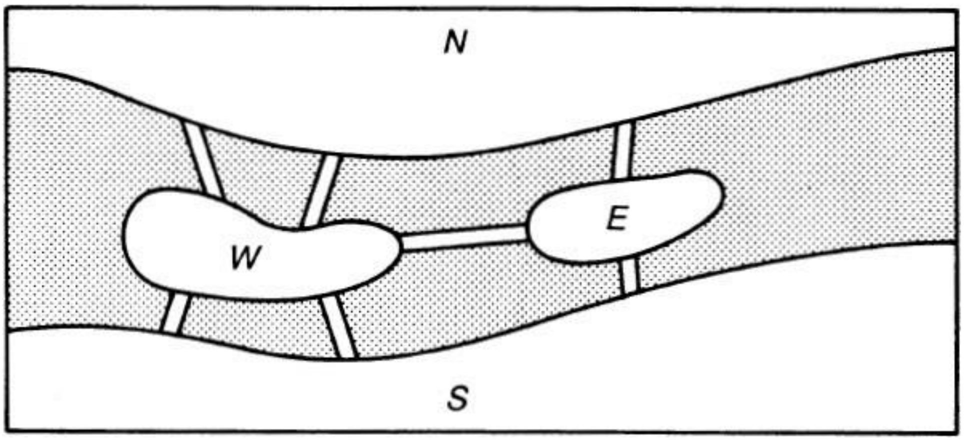
Graf G' = (V', E') adalah subgraf dari G = (V, E) jika V' adalah himpunan bagian dari V dan E' adalah himpunan bagian dari E. Jika W adalah sembarang himpunan bagian dari V, maka subgraf dari G yang diinduksi oleh W adalah graf H = (W, F) di mana f adalah rusuk di F jika f = {u, v}, di mana f ada di E dan u dan v ada di W. Pada Gambar 5.4 , W = {1, 2, 4, 5} adalah himpunan bagian dari himpunan titik V dari graf G dan subgraf dari G yang diinduksi oleh W adalah H. Sebuah subgraf lengkap dari G disebut klik di G.



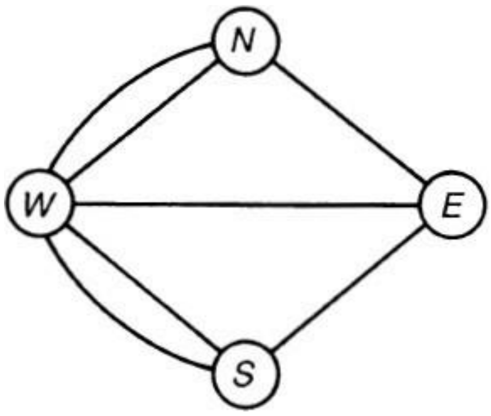
Gambar 5.4 W himpunan bagian dari himpunan titik V dari graf G

Contoh 5.1.1 (Masalah Jembatan Königsberg)

Publikasi pertama dalam teori graf adalah dari Leonhard Euler pada tahun 1736. Makalahnya menyajikan solusi untuk apa yang dikenal sebagai masalah jembatan Königsberg. Kota Konigsberg (sekarang dikenal sebagai Kaliningrad) di Rusia, terletak di tepi Sungai Pregel, terdiri dari pantai utara (N), pantai selatan (S), pulau barat (W), dan pulau timur (E). Menghubungkan keempat bagian ini adalah tujuh jembatan: dua antara N dan W, dua antara S dan W, dan masing-masing satu dari E ke N, S, dan W. (Lihat Gambar 5.5.) Masalah yang diajukan kepada Euler adalah apakah itu mungkin untuk memulai dari lokasi mana pun di kota dan kembali ke titik awal setelah melintasi setiap jembatan tepat satu kali. Jika setiap bagian kota dianggap sebagai simpul dan jika setiap jembatan dianggap sebagai tepi, kita memiliki graf dengan empat simpul dan tujuh sisi (lihat Gambar 5.6), memberikan model graf masalah yang dapat dinyatakan sebagai berikut: Diberikan sebuah graf (tidak harus sederhana), apakah mungkin untuk menelusuri seluruh diagram dari graf tersebut tanpa melewati sisi yang sama lebih dari satu kali? Jawabannya adalah tidak dalam kasus masalah jembatan Konigsberg dengan mudah ditetapkan oleh Euler. Lebih lanjut tentang ini dalam diskusi kami tentang grafik Euler di Bab 6.



Gambar 5.5 Tujuh jembatan: dua antara N dan W, dua antara S dan W, dan E ke N,S dan W



Gambar 5.6 Model graf

Contoh 5.1.2 (Digraf Komunikasi)

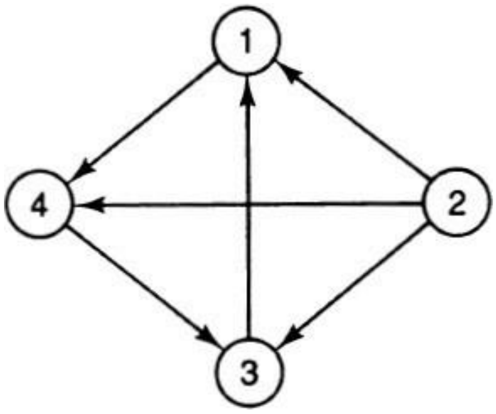
Pertimbangkan sebuah organisasi yang terdiri dari beberapa komponen. Biarkan setiap komponen menjadi simpul. Gambarlah panah dari titik v ke titik w jika komponen v dapat mengirimkan sinyal ke komponen w. Digraf yang dihasilkan dikenal sebagai digraf komunikasi.

Contoh 5.1.3 (Jaringan Transportasi)

Misalkan kita membiarkan setiap simpul dalam grafik mewakili sebuah kota di Amerika Serikat. Dua simpul dihubungkan oleh sebuah sisi jika ada layanan udara langsung nonstop di antara keduanya. Pertanyaan wajar yang muncul adalah apakah mungkin untuk memulai dari sebuah kota dan kembali ke titik awal setelah mengunjungi setiap kota tepat satu kali. Masalah ini dibahas dalam Bab 6 ketika kita menyelidiki graf Hamilton, dinamai dari ahli matematika Irlandia abad kesembilan belas Sir William Hamilton, yang melakukan pekerjaan perintis di bidang ini. Jika bilangan real nonnegatif diberikan pada setiap sisi untuk mewakili biaya penggunaan sisi tersebut, maka masalah optimasi yang terkait adalah menemukan tur semacam itu (jika ada) sehingga biaya tur menjadi sekecil mungkin. Ini adalah masalah travelling salesman problem (TSP) yang terkenal, yang merupakan topik sentral dalam optimasi kombinatorial.

Contoh 5.1.4 (Turnamen)

Dalam turnamen tenis round-robin, setiap pemain harus memainkan setiap pemain lain dan tidak ada ikatan yang diperbolehkan. Biarkan setiap simpul dalam digraf mewakili pemain. Gambarlah panah dari titik v ke titik w jika v mengalahkan w. Digraf yang dihasilkan lengkap dan dikenal sebagai grafik dominasi sebuah turnamen. Digraf dominasi seperti itu sering muncul dalam ilmu sosial dan biologi. Masalah mendasar di sini adalah memutuskan siapa "pemenang" atau "pemimpin" dalam grafik dominasi. Pada Gambar 5.7 kita memiliki turnamen yang terdiri dari empat pemain di mana pemain yang diwakili oleh simpul 2 adalah pemenangnya. Lebih lanjut tentang turnamen di Bab 6.



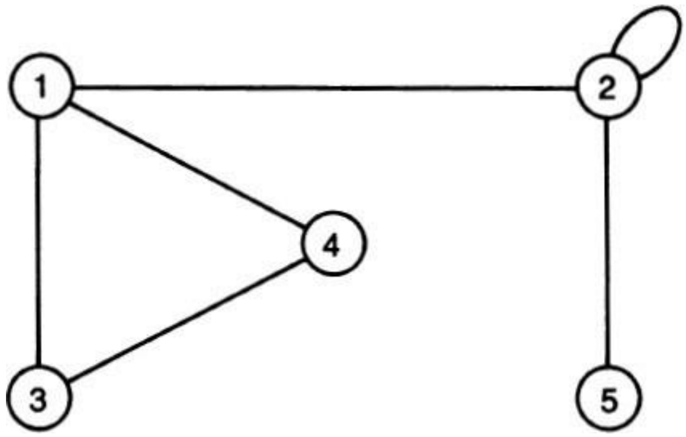
Gambar 5.7 Grafik dominasi

Contoh 5.1.5 (Masalah Penugasan)

Misalkan ada m pelamar kerja P1, p2, . . . , pm dan n pekerjaan q1 q2, . . . , qn. Misalkan V adalah himpunan pelamar kerja dan W himpunan pekerjaan. Jika pi memenuhi syarat untuk qj, tarik tepi antara dua simpul tersebut dan biarkan cij mewakili gaji yang harus dibayarkan kepada p, jika dia dipekerjakan untuk pekerjaan qs. Model yang kita miliki dalam hal ini adalah jaringan bipartit berbobot, dan masalah optimasi kemudian adalah mencari penugasan pekerjaan untuk pelamar sedemikian rupa sehingga (a) semua pekerjaan terisi dan (b) total gaji yang harus dibayar adalah minimum .

5.2 MATRIK ADJACENCY DAN MATRIK INSIDENSI

Untuk tujuan memasukkan grafik ke dalam komputer, perlu untuk menggambarkan grafik tanpa menggunakan representasi piktorialnya. Selain itu, diagram tersebut tidak terlalu praktis ketika grafik dengan sejumlah besar simpul dan tepi yang akan dipelajari. Ada beberapa cara untuk merepresentasikan graf atau digraf tanpa representasi piktorial, dan pada bagian ini akan dibahas beberapa di antaranya. Cara terbaik untuk memasukkan grafik tergantung pada propertinya dan penggunaan selanjutnya. Selanjutnya, efisiensi dari algoritma grafik tergantung pada pilihan metode untuk mewakili grafik yang sedang dipertimbangkan.



Gambar 5.8 Matrik adjacency

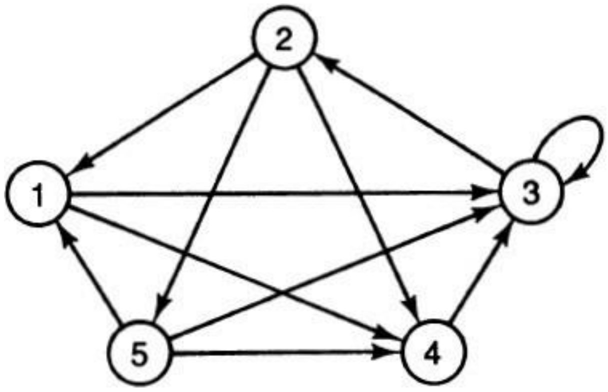
Misalkan G = (V, E) adalah graf tanpa sisi ganda dengan V = {1, 2, 3, . . ., n}. Matriks ketetanggaan dari G adalah matriks n × n A = (aij), dimana aij = 1 jika terdapat rusuk antara simpul i dan simpul j dan aij = 0 sebaliknya. Matriks ketetanggaan suatu graf adalah simetris. Elemen diagonalnya adalah nol jika dan hanya jika tidak ada loop. Matriks ketetanggaan dari graf pada Gambar 5.8 adalah matriks A, dimana:

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Derajat suatu simpul pada suatu graf adalah banyaknya sisi yang datang pada simpul tersebut. Sebuah simpul ganjil jika derajatnya ganjil; jika tidak, itu genap. Pada Gambar 5.8, simpul 1, 2, dan 5 ganjil, dengan derajat 3, 3, dan 1 berturut-turut. Jelasnya, jumlah elemen bukan nol pada baris i dari matriks ketetanggaan suatu graf adalah derajat dari titik i, yang juga sama dengan jumlah semua elemen baris i atau kolom i.

Matriks ketetanggaan suatu digraf dengan n simpul juga merupakan matriks bujur sangkar n × n A = (aij), di mana aij = 1 adalah ada busur dari i ke j dan sebaliknya adalah 0. Matriks ketetanggaan digraf pada Gambar 5.9 adalah A, dimana:

$$A = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

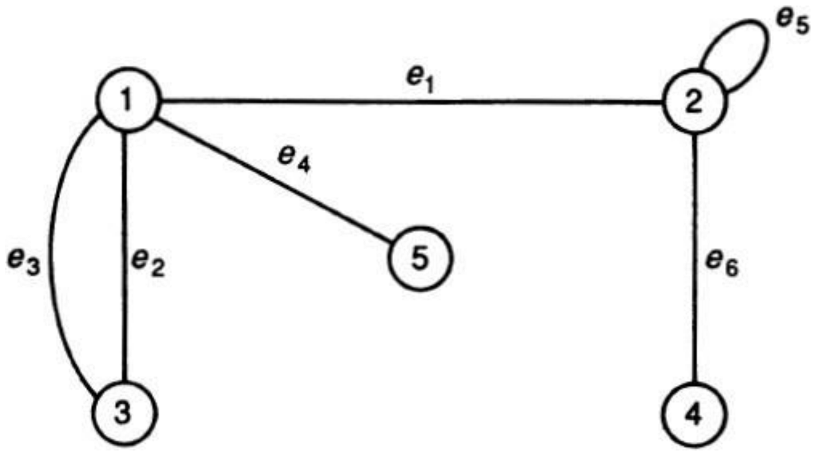


Gambar 5.9 Matriks ketetanggaan

Perhatikan bahwa matriks ketetanggaan suatu digraf tidak harus simetris.

Derajat keluar suatu titik pada digraf adalah jumlah busur yang datang dari titik tersebut dan derajat masuk suatu titik adalah jumlah busur yang datang ke titik tersebut. Pada Gambar 5.9 derajat keluar simpul 2 adalah 3 dan derajat masuknya adalah 1. Perhatikan bahwa jumlah elemen pada baris i adalah derajat keluar dari i dan jumlah elemen pada kolom j adalah derajat masuk j. Perhatikan juga bahwa derajat keluar dari 3 adalah 2 dan derajat masuknya adalah 4.

Matriks lain yang berguna untuk memasukkan grafik dan digraf ke dalam komputer adalah matriks insiden. Berbeda dengan matriks ketetanggaan, matriks kejadian mampu mewakili banyak sisi dan busur paralel. Misalkan G = (V, E) adalah graf di mana V = {1, 2, . . . , n} dan E = {e1 e2, . . . , em}. Matriks kejadian G adalah graf n × m B = (bik), di mana setiap baris bersesuaian dengan sebuah titik sudut dan setiap kolom bersesuaian dengan sebuah rusuk sedemikian rupa sehingga jika ek adalah sebuah rusuk antara i dan j, maka semua elemen kolom k adalah 0 kecuali bik = bjk = 1. Misalnya, matriks kejadian dari grafik pada Gambar 5.10 adalah B, di mana:



Gambar 5.10 Matrik yang berguna memasukkan grafik dan digraf

Perhatikan bahwa kolom yang sesuai dengan tepi memiliki tepat dua elemen bukan nol jika itu bukan loop dan tepat satu elemen bukan nol jika itu adalah loop. Selanjutnya, jumlah elemen baris i adalah derajat simpul i.

Kita juga mengamati bahwa dalam sembarang graf tanpa loop, jumlah derajat semua simpul adalah dua kali jumlah rusuk karena setiap rusuk dihitung dua kali, satu kali untuk setiap simpul datangnya. Sebagai contoh, pada Gambar 4.1.6 kita melihat derajat N + derajat S + derajat W + derajat E = 3 + 3 + 5 + 3 = 14 = dua kali jumlah sisi. Kami menyatakan properti ini sebagai teorema yang kadang-kadang dikenal sebagai teorema pertama dari teori graf.

TEOREMA 5.2.1

Jika G adalah multigraf tanpa loop dan m sisi, jumlah derajat semua simpul G adalah 2m.

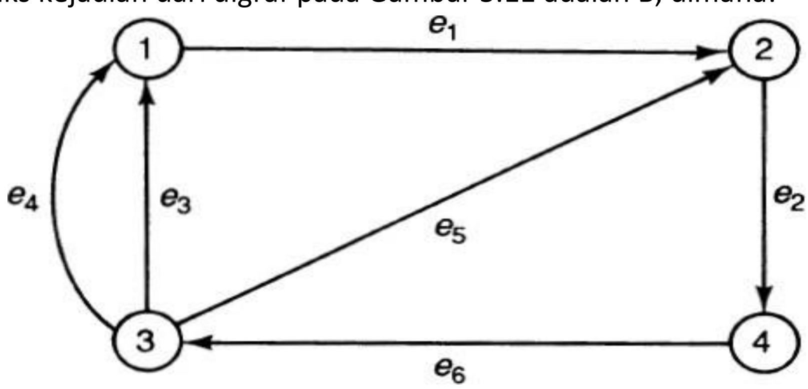
Akibat Wajar

Banyaknya simpul ganjil pada multigraf loopless adalah genap.

Bukti:

Misalkan jumlah simpul ganjil adalah r. Misalkan p adalah jumlah derajat semua simpul ganjil dan q adalah jumlah derajat semua simpul genap. Maka p + q genap dengan teorema. Juga, q genap, jadi p genap. Tetapi p adalah jumlah r bilangan ganjil. Oleh karena itu, r genap, yang membuktikan pernyataan tersebut.

Matriks kejadian B suatu digraf (tanpa loop) didefinisikan sebagai berikut: Jika ek adalah busur dari i ke j, semua elemen pada kolom k adalah nol kecuali bik = – 1 dan bjk = 1. Misalnya, matriks kejadian dari digraf pada Gambar 5.11 adalah B, dimana:



Gambar 5.11 Matriks kejadian dari digraf

$$
\begin{array}{cc}
 & \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\
B = & \begin{bmatrix} -1 & 0 & 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & -1 & 1 \\ 0 & 1 & 0 & 0 & 0 & -1 \end{bmatrix}
\end{array}
$$

Perhatikan bahwa jumlah semua elemen pada baris i dari matriks kejadian suatu digraf sama dengan derajat masuk i dikurangi derajat keluar i. Kami juga mengamati bahwa dalam setiap digraf jumlah semua derajat keluar sama dengan jumlah total busur, yang sekali lagi sama dengan jumlah semua derajat masuk. Ini karena ketika derajat keluar dijumlahkan, setiap busur dihitung satu kali karena setiap busur datang dari satu titik. Demikian pula, ketika indegrees dijumlahkan, setiap busur dihitung satu kali karena setiap busur bersinggungan dengan satu simpul.

5.3 BERGABUNG DALAM GRAFIK

Lintasan antara dua simpul v1 dan vr dalam suatu graf adalah barisan berhingga dari simpul dan sisi yang berbentuk v1, e1 v2, e2, v3, e3, . . . , er, vr, di mana ek adalah tepi antara vk -1 dan vk . Secara umum, simpul dan tepi dalam suatu jalur tidak perlu berbeda.

Suatu lintasan dikatakan sederhana jika simpul-simpulnya berbeda. Dalam jalur sederhana, jelas semua tepinya berbeda. Tapi jalan dengan tepi yang berbeda dapat memiliki simpul berulang. Suatu graf dikatakan terhubung jika terdapat lintasan antara setiap pasangan simpul di dalamnya.

Lintasan antara simpul dan dirinya sendiri adalah lintasan tertutup. Lintasan tertutup yang semua sisinya berbeda adalah sirkuit. Sirkuit di mana semua simpulnya berbeda adalah siklus. Perhatikan bahwa v, e1, w, e2, v adalah siklus tetapi v, e, w, e, v bukan sirkuit dan karenanya bukan siklus. Kedua jalur tertutup ini dapat direpresentasikan sebagai:

$$v \overset{e_1}{- - - - -} w \overset{e_2}{- - - - -} v \qquad \text{and} \qquad v \overset{e}{- - - - -} w \overset{e}{- - - - -} v$$

Pada Gambar 5.12, 1 - - - - 2 - - - - 3 - - - - 2 - - - - 1 - - - - 5 adalah lintasan dan 1 - - - - 2 - - - - 3 - - - - 2 - - - - 1 - - - -5 adalah jalur sederhana, dan kita juga dapat melihatnya

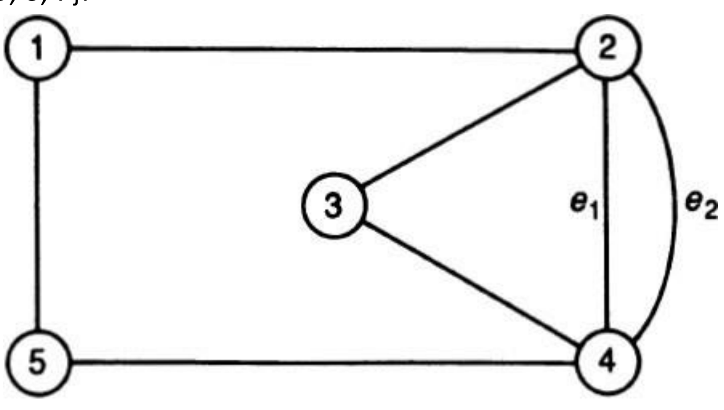
$$2 - - - - 3 - - - - - 4 - - - - 5 - - - - 1 - - - - 2 \overset{e_1}{- - - -} 4 \overset{e_2}{- - - -} 2$$

adalah sirkuit dan

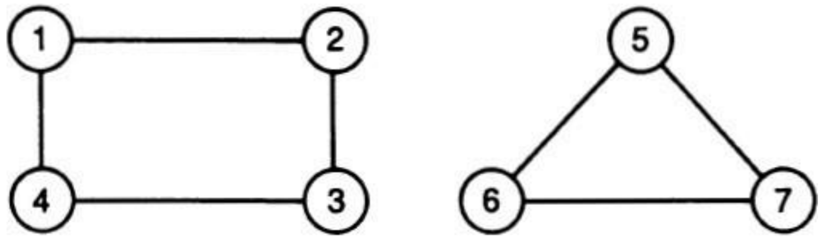
$$2 \overset{e_1}{- - - -} 4 - - - - - 3 - - - - - 2$$

adalah sebuah siklus.

Jika v dan w terhubung (yaitu, ada jalur di antara keduanya), maka w dan v terhubung. Sebenarnya, relasi J yang didefinisikan oleh vJw jika v dan w terhubung adalah relasi ekivalensi yang mempartisi himpunan V dari simpul-simpul menjadi himpunan bagian lepas berpasangan dari V. Subgraf yang diinduksi oleh setiap subset tersebut adalah subgraf terhubung maksimal yang disebut komponen dari graf . Banyaknya komponen dari suatu graf G dilambangkan dengan K(G) dan sama dengan 1 jika dan hanya jika G terhubung. Grafik pada Gambar 5.13 memiliki dua komponen, G′ dan G′′, dimana G′ diinduksi oleh himpunan bagian {1, 2, 3, 4} dan G″ diinduksi oleh {5, 6, 7}.



Gambar 5.12 Lintasan jalur sederhana



Gambar 5.13 Grafik dengan dua komponen,G' dan G''

Ada hubungan yang menarik dan berguna antara jumlah jalur antara pasangan simpul di G dan elemen pangkat dari matriks ketetanggaannya. Jalur dengan k sisi disebut k-path. 1-path adalah edge. Dalam matriks ketetanggaan n × n dari graf (tanpa sisi ganda) dengan himpunan simpul V = {1, 2, . . . , n}, elemen (i, j) adalah 1 jika dan hanya jika jumlah jalur 1 (sisi) antara i dan j adalah 1. Bahwa hasil ini dapat digeneralisasi adalah isi dari pernyataan berikut.

TEOREMA 5.3.1

Jika A adalah matriks ketetanggaan dari suatu graf, entri (i, j) dari pangkat ke-k (k 1) dari A adalah jumlah jalur-k antara simpul i dan simpul j.

Bukti:

Pembuktiannya dengan induksi pada k. Ini benar ketika k = 1. Asumsikan bahwa ini benar untuk (k – 1). Membiarkan $a_{ij}^{(r)}$ menjadi (i, j)- entri dari kekuatan r dari A. Kemudian:

$$a_{ij}^{(k)} = \sum_{p=1}^{p=n} a_{ip}^{(k-1)} a_{pj} \qquad \text{since } A^k = A^{k-1} \cdot A \qquad (*)$$

tapi,

$$a_{ip}^{(k-1)} a_{pj} = \begin{cases} a_{ip}^{(k-1)} & \text{if } p \text{ and } j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

Dengan hipotesis, $a_{ip}^{(k-1)} = \text{number of } (k-1)\text{-paths}$ antara i dan p, dan ini akan sama dengan jumlah k-path antara dari i ke j, di mana vertex sebelum j adalah p. Jadi ruas kanan (*) adalah jumlah total k-path antara i dan j yang diperoleh setelah memeriksa p = 1 sampai p = n berturut-turut.

Akibat Wajar

(i, i) – entri di A2 adalah derajat i.

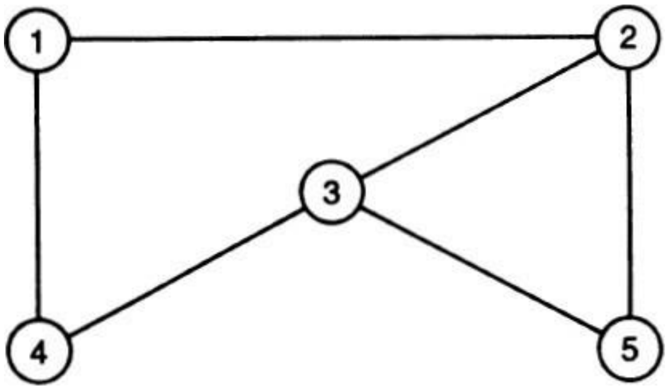
Contoh 5.3.1

Sebagai ilustrasi perhatikan matriks A, A,2 dan A4 dari grafik Gambar 5.14. Kita punya:

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \quad A^2 = \begin{bmatrix} 2 & 0 & 2 & 0 & 1 \\ 0 & 3 & 1 & 2 & 1 \\ 2 & 1 & 3 & 0 & 1 \\ 0 & 2 & 0 & 2 & 1 \\ 1 & 1 & 1 & 1 & 2 \end{bmatrix} \quad A^4 = \begin{bmatrix} 9 & 3 & 11 & 1 & 6 \\ 3 & 15 & 7 & 11 & 8 \\ 11 & 7 & 15 & 3 & 8 \\ 1 & 11 & 3 & 9 & 6 \\ 6 & 8 & 8 & 6 & 8 \end{bmatrix}$$

Dalam A2 entri (4, 4) adalah 2 dan derajat simpul 4 adalah 2 dan dua jalur 2 antara 4 dan 4 adalah 4 - - - - 1 - - - -4 dan 4 - - - - 3 - - - - 4. Dari pangkat empat A kita melihat bahwa ada delapan 4-jalur yang berbeda antara 2 dan 5.

Akhirnya, tiga definisi lagi: Tepi dalam graf terhubung disebut jembatan jika penghapusan tepi itu, tetapi bukan simpul ujungnya, membuat graf terputus. Graf tanpa siklus adalah graf asiklik, disebut juga hutan. Hutan yang terhubung adalah pohon. Kami mempelajari pohon secara rinci di Bab 6 dan 7.



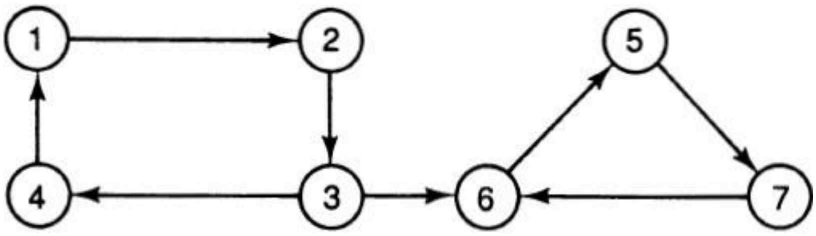
Gambar 5.14 Grafik matriks A, A^2, dan A^4

5.4 MENJANGKAU DALAM DIGRAF

Lintasan berarah dari simpul v ke simpul w dalam digraf adalah barisan berhingga v1, a1 v2, a2, , vr, ar, vr+1 dari simpul dan busur, di mana simpul pertama adalah v dan simpul terakhir adalah w dan ai adalah busur dari vi ke vi+1. Jika ada lintasan berarah dari v ke w, maka v terhubung ke w dan w terhubung dari v. Sepasang simpul adalah pasangan terhubung kuat jika masing-masing terhubung satu sama lain. Jika salah satu dari mereka terhubung ke yang lain, itu adalah pasangan yang terhubung secara sepihak. Digraf sangat kuat terhubung jika setiap pasangan simpul adalah pasangan terhubung kuat dan terhubung secara sepihak jika setiap pasangan terhubung secara sepihak. Suatu digraf terhubung lemah jika graf di bawahnya terhubung.

Lintasan berarah dari suatu simpul ke dirinya sendiri adalah lintasan berarah tertutup. Lintasan berarah tertutup adalah rangkaian berarah jika busurnya berbeda dan merupakan siklus berarah jika simpulnya berbeda. Perhatikan perbedaan halus antara grafik dan digraf: Jika simpul dari jalur berarah tertutup berbeda, busurnya berbeda. Tetapi jika simpul-simpul dari suatu lintasan tertutup dalam suatu graf berbeda, maka sisi-sisinya tidak perlu berbeda.

Relasi yang didefinisikan oleh vRw jika {v, w} adalah pasangan terhubung kuat adalah relasi ekivalensi yang memberikan partisi dari himpunan titik V ke dalam kelas himpunan bagian berpasangan yang saling lepas, dan subgraf yang diinduksi oleh salah satu dari himpunan bagian ini disebut kuat komponen digraf. Misalnya, dalam digraf Gambar 5.15 kita memiliki dua komponen kuat yang diinduksi oleh himpunan {1, 2, 3, 4} dan {5, 6, 7}. Seperti dalam kasus graf, elemen pangkat ke-k dari matriks ketetanggaan A dari digraf dapat digunakan untuk menghitung jumlah k-path antara pasangan simpul. Bukti dari hasil berikut dibiarkan sebagai latihan.



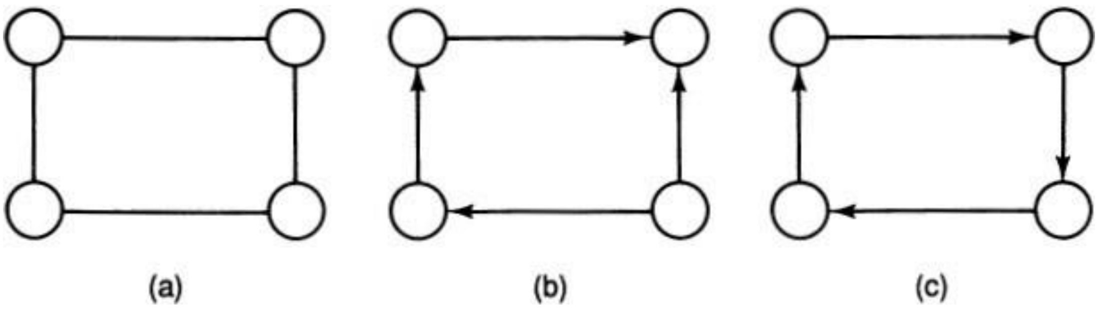
Gambar 5.15 Digraf dengan dua komponen kuat

TEOREMA 5.4.1

Jika A adalah matriks ketetanggaan suatu digraf, maka (i, j)-entri pangkat ke-k (k 1) dari A adalah jumlah lintasan berarah k dari i ke j.

Jika G adalah graf, maka digraf G′ yang diperoleh dari G dengan mengubah setiap sisi G menjadi busur disebut orientasi G. Sebagai contoh, pada Gambar 5.16 digraf (b) dan (c) keduanya orientasi grafik (a).

Orientasi graf disebut orientasi kuat dari graf jika orientasinya terhubung kuat. Pada Gambar 5.16 digraf pada (c) merupakan orientasi kuat dari graf pada (a). Suatu graf dikatakan sangat berorientasi jika memiliki orientasi yang kuat. Sangat mudah untuk melihat bahwa graf yang sangat berorientasi harus terhubung dan tanpa jembatan. Kebalikannya juga berlaku: Graf sangat berorientasi jika dan hanya jika terhubung dan tanpa jembatan. Teorema ini disebabkan oleh H. E. Robbins (1939) dan kami menghilangkan buktinya. Kemudian dalam bab ini kita membahas algoritma untuk mendapatkan orientasi yang kuat dari grafik jika orientasi seperti itu ada.



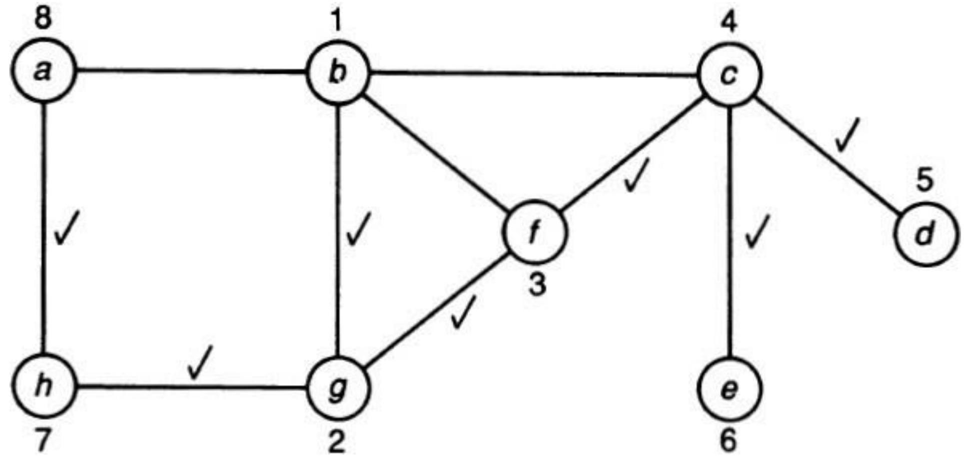
Gambar 5.16 Digraf (b) dan (c) keduanya orientasi grafik (a)

Matriks reachability dari digraf dengan n simpul adalah matriks n × n R = (rij), di mana rij adalah 1 jika ada jalur berarah dari i ke j, dan 0 sebaliknya. Jelas, sebuah digraf terhubung kuat jika dan hanya jika setiap elemen matriks reachabilitynya sama dengan 1.

5.5 MENGUJI KETERHUBUNGAN

Mengingat grafik, wajar untuk bertanya apakah itu terhubung. Tentu saja, dari diagram graf seseorang dapat dengan mudah melihat apakah graf tersebut memiliki lebih dari satu komponen dan dengan demikian menguji keterhubungannya. Untuk grafik besar, diagram seperti itu tidak layak. Selain itu, jika kita memasukkan grafik ke dalam komputer, kita membutuhkan algoritma untuk melihat apakah grafik itu terhubung. Salah satu algoritma tersebut adalah teknik depth-first search (DFS), di mana kita memberi label ulang simpul-simpul grafik sebagai berikut.

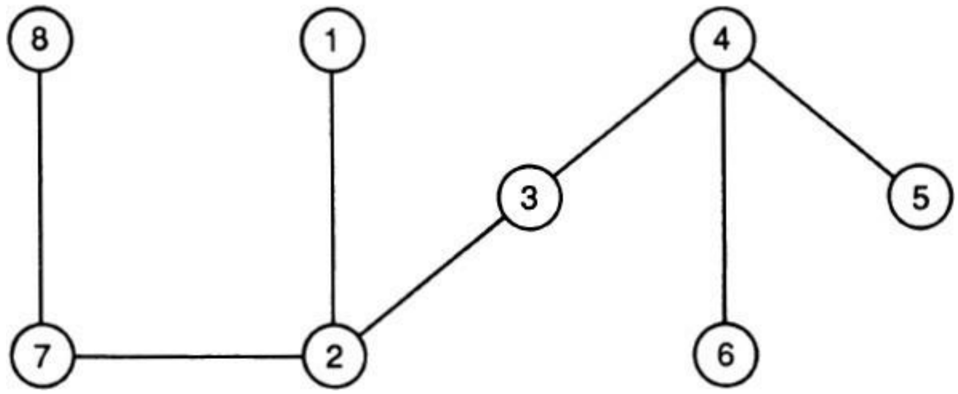
Biarkan simpul dari grafik G menjadi v1, v2, . . . , dan vn. Pilih sembarang simpul dan beri label sebagai 1. Pilih sembarang simpul yang berdekatan dengan 1. Ini belum diberi label; beri label 2. Tandai tepi {1, 2} sebagai tepi bekas agar tidak digunakan lagi. Lanjutkan dengan cara yang sama, misalkan kita memberi label simpul vi dengan bilangan bulat k. Cari di antara semua simpul berdekatan yang tidak berlabel dari simpul ini, pilih salah satunya, dan beri label sebagai (k + 1). Tandai tepi {k, k + 1} sebagai tepi bekas. Sekarang mungkin kasus bahwa semua simpul yang berdekatan dari k diberi label. Jika demikian, kembalilah ke simpul (k – 1) dan cari di antara simpul-simpul berdekatan yang tidak berlabel. Jika kita menemukan salah satu simpul tersebut, beri label sebagai (k + 1) dan tandai tepi {k – 1, k + 1} sebagai tepi bekas. Lanjutkan proses sampai semua simpul diberi label atau kita kembali ke simpul 1 dengan setidaknya satu simpul tidak berlabel. Dalam kasus sebelumnya grafik terhubung dan akan ada tepat (n – 1) tepi yang digunakan. Subgraf asiklik yang terdiri dari n simpul dari graf dan (n – 1) tepi yang digunakan ini disebut pohon rentang pencarian kedalaman-pertama dari graf. Jika tidak mungkin untuk melabeli semua n simpul dengan teknik DFS, kita simpulkan bahwa graf tidak terhubung. Prosedur serupa untuk menguji keterkaitan kuat digraf diberikan dalam Tarjan (1971).



Gambar 5.17 Tuju tepi ditandai dari pohon rentang DFS

Mari kita ilustrasikan teknik pelabelan DFS ini pada grafik Gambar 5.17 dengan delapan simpul a, b, c, d, e, f, g, dan h. Kita pilih simpul b dan beri label 1. Sebuah simpul yang berdekatan dengan 1 adalah g. Beri label sebagai 2. Tandai tepi {1, 2} sebagai tepi bekas. Sebuah simpul yang berdekatan dengan 2 dan tidak berlabel adalah f, dan berlabel 3. Pada tahap ini, tepi {2, 3} ditandai. Kita sekarang melihat bahwa c adalah satu-satunya simpul tak berlabel yang berdekatan dengan 3. Jadi c dilabeli sebagai 4 dan {3, 4} ditandai. Kemudian d atau e dapat diberi label 5. Ikatan diputus dengan memberi label d sebagai 5 dan {4, 5} ditandai. Kita perhatikan bahwa 5 tidak memiliki simpul bertetangga yang tidak berlabel, jadi kita kembali ke 4 dan memberi label e sebagai 6. Pada tahap ini kita kembali ke 4, lalu ke 3, dan kemudian ke 2 untuk mencari simpul bertetangga yang tidak berlabel. Kami memberi label h sebagai 7 dan a sebagai 8 dan menandai tepi {2, 7} dan {7, 8}. Pada titik ini kedelapan simpul tersebut diberi label, yang menunjukkan bahwa graf tersebut memang terhubung. Tujuh tepi yang ditandai adalah tepi dari pohon rentang DFS seperti yang ditunjukkan pada Gambar 5.18.

Sekarang mari kita cari kompleksitas komputasi dari algoritma DFS menggunakan contoh yang baru saja dibahas. Perhatikan bahwa setiap sisi {i, j}, di mana i < j dapat diselidiki dalam arah maju dari i ke j atau dalam arah mundur dari j ke i. Dalam contoh kami, kami menyelidiki {1, 2}, {2, 3}, {3, 4}, dan {4, 5} ke arah depan. Karena 5 tidak memiliki simpul bertetangga yang tidak berlabel, kami harus kembali ke 4, yang berarti bahwa kami harus menyelidiki {4, 5} dalam arah mundur. Jadi tepi {4, 5} diperiksa dua kali dan tidak akan pernah diselidiki lagi. Jadi setiap tepi diperiksa paling banyak dua kali. Jadi jika ada m sisi, akan ada paling banyak 2m penyelidikan, dan ada n simpul yang akan diberi label. Jadi kompleksitasnya paling banyak n + 2m. Karena nilai maksimum untuk m adalah n(n – l)/2, kompleksitas kasus terburuk dari algoritma DFS adalah n2.



Gambar 5.18 Pohon rentang DFS

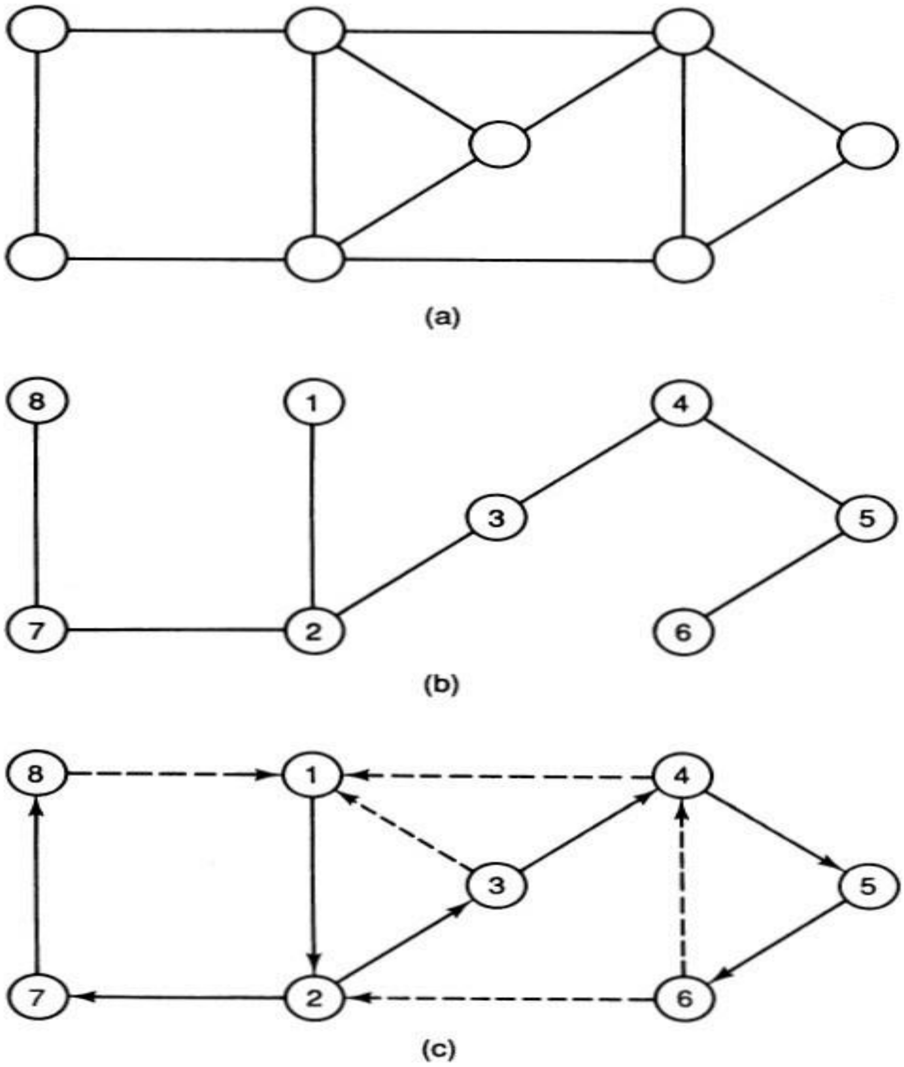
5.6 ORIENTASI GRAFIK YANG KUAT

Pertimbangkan model graf di mana simpulnya adalah sudut jalan kota besar. Dua simpul dihubungkan oleh sebuah sisi jika ada jalan yang menghubungkannya. Misalkan grafik G yang dihasilkan sangat berorientasi. Ini adalah kasus jika dan hanya jika G terhubung dan tanpa jembatan. Kami sekarang tertarik untuk (sementara) mengubah semua jalan di kota menjadi jalan satu arah. Karena G sangat berorientasi, setiap sudut dapat dicapai dari setiap sudut lainnya setelah konversi ini. Bagaimana konversi ini dilakukan? Kami kembali menggunakan prosedur DFS dan memberi label semua simpul. Jika {i, j} adalah sisi bertanda di mana i < j, ubah sisi ini menjadi busur dari i ke j. Di sisi lain, jika {i, j} adalah sisi yang tidak bertanda di mana i < j, ubahlah sisi ini menjadi busur dari j ke i. Digraf yang dihasilkan G′ adalah orientasi yang kuat dari G. Untuk bukti pernyataan ini, lihat Roberts (1976). Pada Gambar 5.19 kita memiliki graf tanpa jembatan yang terhubung di (a), dengan pohon rentang DFS di (b), di mana simpul diberi label dengan tepat. Orientasi kuat G adalah digraf G′ dari Gambar 5.19(c). Ketika kita menggunakan prosedur DFS akan ada m tepi yang tidak ditandai dalam kasus terburuk yang harus diselidiki. Jadi kompleksitasnya paling banyak n + 2m + m, yang dalam kasus terburuk akan sama dengan f(n) = (3n2 – n)/2.

5.7 CATATAN DAN REFERENSI

Ada beberapa referensi yang sangat baik tentang teori graf baik di tingkat pengantar maupun di tingkat lanjutan. Berikut adalah sebagian daftarnya: Behzad et al. (1979), Berge (1962), Bondy dan Murty (1976), Carre (1979), Chartrand (1977), Deo (1974), Gibbons (1985), Gondran dan Minoux (1984), Harary (1969a), Ore (1963), Roberts (1976, 1978), Swamy dan

Thulasiraman (1981), Wilson (1979), dan Yemelichev, dkk. (1984). Bab-bab tentang grafik dalam buku-buku karya Grimaldi (1985), Liu (1985), Roberts (1984), Townsend (1987), dan Tucker (1984) juga sangat direkomendasikan. Untuk bukti teorema Robbins, lihat Bab 2 Roberts (1978), yang juga berisi diskusi lengkap tentang orientasi kuat dan penugasan jalan satu arah.



Gambar 5.19 Graf tanpa jembatan

5.8 LATIHAN

1. Gambarlah grafik G = (V, E), dengan V = {1, 2, 3, 4, 5} dan E = {{1, 2}, {1, 3}, {1, 5}, {2, 3}, {3, 4}, {3, 5}, {4, 5}}. Tentukan himpunan W = {i : i adalah simpul sedemikian sehingga i dan 2 bertetangga}.
2. Gambarlah digraf yang graf dasarnya adalah graf Soal 4.1. Temukan himpunan simpul (a) yang bertetangga dengan simpul 2 dan (b) bertetangga dari simpul 2 pada digraf ini.
3. (a) Buatlah graf lengkap dengan empat simpul sehingga tidak ada dua sisi yang berpotongan, (b) Buatlah graf lengkap dengan lima simpul, (c) Apakah Anda memperhatikan perbedaan antara graf pada bagian (a) dan (b)?
4. Tentukan jumlah rusuk pada graf lengkap dengan n simpul.

5. Gambarlah grafik transportasi udara dengan Boston, New York, London, Paris, Moskow, Praha, dan Roma sebagai simpul dan sisi yang menghubungkan dua kota jika ada layanan udara nonstop di antara mereka.
6. Gambarlah subgraf yang diinduksi oleh V′ = {2, 3, 4, 5} pada Soal 4.1.
7. Tentukan jumlah minimum jembatan (a) yang akan dibangun dan (b) yang akan dibongkar sehingga masalah jembatan Königsberg dapat diselesaikan.
8. (a) Gambarkan graf bipartit K2,2 sehingga tidak ada dua sisi yang berpotongan, (b) Gambarkan graf bipartit K3.3. (c) Apa perbedaan mencolok antara kedua graf bipartit ini?
9. Tentukan jumlah rusuk dalam Kp,q.
10. Perhatikan graf G = (V, E) dengan V = {1, 2, 3, 4, 5} dan E = {a, b, c, d, e, f, g, h}, di mana a = {1 , 2}, b = {2, 3}, c = {3, 5}, d = {2, 5}, e = {2, 4}, f = {4, 5}, g = {1, 4 }, dan h = {1, 5}.
    (a) Carilah matriks ketetanggaan dari G.
    (b) Temukan derajat setiap simpul dan temukan himpunan simpul ganjil.
    (c) Carilah matriks kejadian dari G.
11. Perhatikan sebuah digraf G′ yang graf dasarnya adalah graf G dari Soal 4.1.
    (a) Carilah matriks ketetanggaan dari G′.
    (b) Tentukan derajat masuk dan derajat tidak derajat masing-masing simpul di G′.
    (c) Carilah matriks kejadian dari G′.
12. Berikan contoh graf sederhana dengan (a) tidak ada simpul ganjil, (b) tanpa simpul genap.
13. Buatlah graf sederhana terhubung dengan n simpul sedemikian rupa sehingga derajat setiap simpulnya adalah 2. Perhatikan struktur dari graf tersebut.
14. Buatlah graf dengan n simpul dan (n – 1) rusuk sedemikian rupa sehingga terdapat dua simpul berderajat 1 dan (n – 2) simpul berderajat 2.
15. Graf G dengan sifat semua simpulnya berderajat r sama disebut graf beraturan berderajat r. Perhatikan bahwa suatu graf lengkap beraturan tetapi kebalikannya tidak benar, (a) Buatlah graf beraturan sederhana berderajat 1 yang tidak lengkap, (b) Bangunlah graf beraturan sederhana berderajat 2 yang tidak lengkap, (c) Jika G adalah graf beraturan berderajat r dan jika G memiliki n simpul, tentukan jumlah rusuk di G.
16. Buktikan bahwa pada graf sederhana dengan dua atau lebih simpul, derajat dari simpul-simpul tersebut tidak dapat berbeda semua.
17. Misalkan G adalah graf sederhana dengan n simpul dengan A sebagai matriks ketetanggaannya dan B sebagai matriks kejadiannya. Tentukan n × n matriks diagonal C di mana elemen diagonal ke-i adalah derajat simpul i di G. C disebut matriks derajat G. Buktikan bahwa B • Bt = A + C.
18. Matriks bujur sangkar yang setiap elemennya 0 atau 1 disebut matriks dominan jika (a) setiap bilangan diagonalnya adalah 0 dan (b) elemen (i, j) adalah 1 jika dan hanya jika elemen (j, i) adalah 0. Buktikan bahwa matriks ketetanggaan suatu turnamen adalah matriks dominasi.
19. Perhatikan digraf G = (V, E) dimana V = {1, 2, 3, 4, 5, 6} dan E = {(1, 2), (2, 3), (3, 4), (4 , 5), (5, 6), (1, 6), (2, 6), (5, 2)}.
    (a) Temukan 6-jalur dari 1 hingga 6.

(b) Temukan jalur sederhana dari 1 hingga 6 menggunakan lima busur.

(c) Carilah sebuah lingkaran dengan empat busur.

(d) Gunakan matriks adjacency dari G untuk menentukan jumlah 2-jalur dari 2 hingga 4.

(e) Temukan semua komponen kuat dari G.

(f) Carilah matriks reachability R dari G.

20. Biarkan R menjadi matriks reachability dari digraf G = (V, E) di mana V = {1, 2, . . . , n}, P = (pij) adalah hasil kali elemen R dan transpos R, dan Q = R2 = (qij). Membuktikan:

    (a) pij = 1 jika dan hanya jika i dan j terhubung kuat

    (b) qii = jumlah simpul dalam komponen kuat yang memuat simpul i.

21. Buatlah graf dengan lima simpul dan enam sisi yang terdiri dari sirkuit dengan enam sisi dan dua siklus dengan masing-masing tiga sisi.

22. Perhatikan grafik Soal 4.1.

    (a) Temukan 6-jalur antara 1 dan 4.

    (b) Temukan jalur sederhana antara 1 dan 4 dengan empat sisi.

    (c) Gunakan matriks ketetanggaan untuk menentukan jumlah jalur 2 antara 2 dan 4.

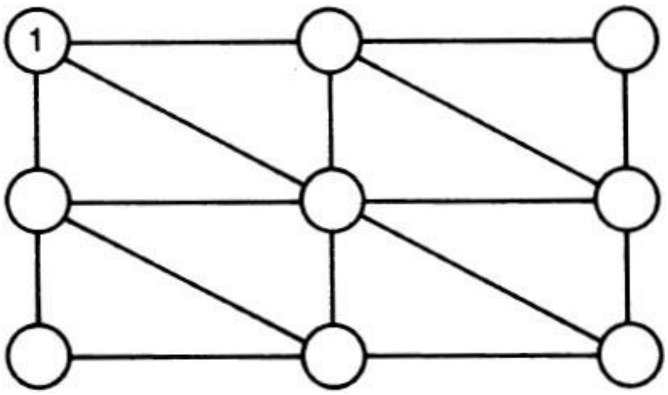
23. Tentukan matriks reachability dari grafik.

24. Gambarlah graf G dengan matriks ketetanggaan A sedemikian sehingga

$$A^2 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 3 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} \quad \text{and} \quad A^3 = \begin{bmatrix} 0 & 3 & 1 & 1 \\ 3 & 2 & 4 & 4 \\ 1 & 4 & 2 & 3 \\ 1 & 4 & 3 & 2 \end{bmatrix}$$

25. Tunjukkan bahwa jumlah elemen diagonal pangkat dua dari matriks ketetanggaan suatu graf G adalah dua kali jumlah rusuk di G.

26. Jika G adalah graf terhubung dengan n simpul, tunjukkan bahwa terdapat lintasan dengan paling banyak (n – 1) rusuk di antara setiap pasangan simpul.



Gambar 5.20 Matriks ketetanggaan

27. Jika A adalah matriks ketetanggaan dari graf dengan n simpul dan jika elemen nondiagonal dari A + A2 + A3 + • • + An–1 adalah nol, apa yang dapat Anda katakan tentang G?

28. Dapatkan pohon merentang DFS (dimulai dari simpul 1) pada graf terhubung G pada Gambar 4.8.1.
29. Graf G pada Soal 4.28 terhubung dan tanpa jembatan. Temukan orientasi yang kuat dari G.
30. Segitiga dikatakan monokromatik jika semua sisinya berwarna sama. Tunjukkan bahwa tidak peduli bagaimana kita mewarnai tepi dari graf lengkap dengan enam simpul menggunakan dua warna, akan selalu ada setidaknya satu segitiga monokromatik. (Lihat Contoh 1.5.5.) Dapat ditunjukkan bahwa setidaknya ada dua segitiga seperti itu. Tunjukkan juga bahwa adalah mungkin untuk mewarnai semua sisi dari suatu graf lengkap dengan lima titik dengan menggunakan dua warna sedemikian rupa sehingga tidak ada segitiga monokromatik.

BAB 6
LEBIH LANJUT TENTANG GRAFIK DAN DIGRAF

6.1 JALUR EULERIAN DAN SIRKUIT EULERIAN

Lintasan dalam suatu graf adalah lintasan Euler jika setiap sisi pada graf muncul sebagai sisi pada lintasan tepat satu kali. Lintasan Euler tertutup adalah lintasan Euler. Suatu graf dikatakan graf Euler jika memiliki sirkuit Euler. Ada definisi analog dalam kasus digraf.

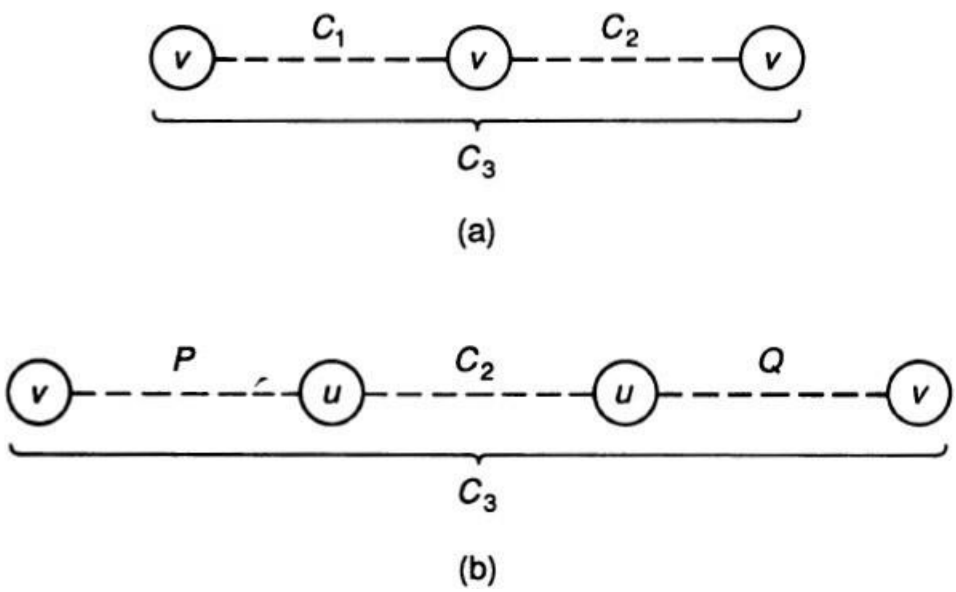
Gagasan sirkuit Euler pertama kali muncul dari masalah jembatan Konigsberg yang terkenal (Contoh 5.1.1), yang menanyakan apakah seseorang dapat melintasi ketujuh jembatan di kota, melewati setiap jembatan tepat satu kali, dan kembali ke lokasi awal. Dalam rangka menunjukkan bahwa tidak mungkin untuk melakukannya, Euler menghasilkan teknik yang, diyakini secara universal, melahirkan teori graf. Jelas bahwa masalah dapat diselesaikan jika model grafnya (lihat Gambar 5.6) adalah graf Euler. Teorema berikut menyelesaikan pertanyaan ini.

TEOREMA 6.1.1

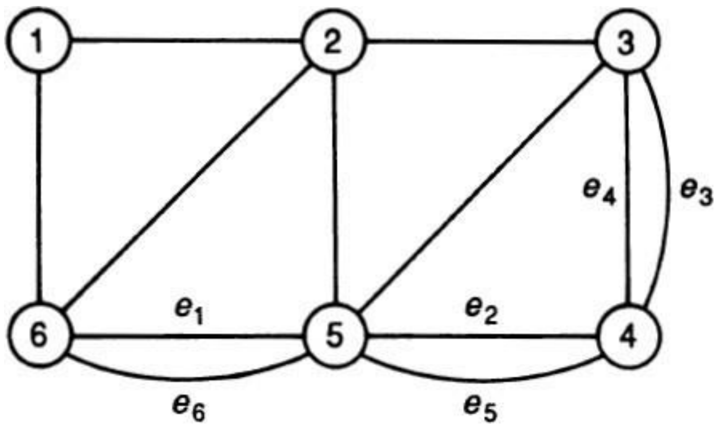
Graf terhubung G tanpa loop adalah Euler jika dan hanya jika derajat setiap simpul genap.

Bukti:

Setiap sirkuit Euler di G meninggalkan setiap simpul sebanyak yang ia masuki. Jadi setiap simpul dari G genap. Sebaliknya, misalkan G adalah graf terhubung yang setiap simpulnya genap. Kami membuktikan bahwa G adalah Euler dengan benar-benar membangun sirkuit Euler di dalamnya. Ada beberapa algoritma untuk konstruksi ini. Untuk detailnya, lihat Genap (1979). Kami mengadopsi prosedur berikut, di mana sirkuit "disambung," sampai kami benar-benar mendapatkan sirkuit Euler. Mulai dari sembarang simpul v. Lintasi sisi-sisi yang berbeda dari G sampai kita kembali ke v. Hal ini tentu saja mungkin karena setiap simpul genap. Biarkan C1 menjadi sirkuit yang diperoleh. Jika sirkuit ini berisi semua tepi grafik, kita selesai. Jika tidak, hapus semua tepi sirkuit ini dan semua simpul berderajat 0 dari G untuk mendapatkan subgraf terhubung H1 di mana setiap simpul juga genap. Selanjutnya, karena G terhubung, ada simpul u yang sama untuk sirkuit C1 dan subgraf H1. Sekarang mulai dari u dan dapatkan rangkaian C2 dengan melintasi tepi subgraf yang berbeda. Perhatikan bahwa kedua sirkuit tidak memiliki tepi yang sama, meskipun mereka mungkin memiliki simpul yang sama. Jika v = u, kedua sirkuit dapat digabungkan untuk membentuk sirkuit C3 yang diperbesar. Lihat Gambar 6.1(a). Jika v dan u berbeda, misalkan P dan Q adalah dua jalur sederhana yang berbeda antara v dan u yang terdiri dari sisi-sisi dari C1 . Kemudian P, Q, dan C2 disambung bersama untuk membentuk rangkaian baru C3, seperti pada Gambar 6.1(b). Jika sirkuit yang diperbesar ini memiliki semua sisi G, kita simpulkan bahwa itu adalah Euler. Jika tidak, kita lanjutkan sampai diperoleh rangkaian yang memiliki semua sisi G.



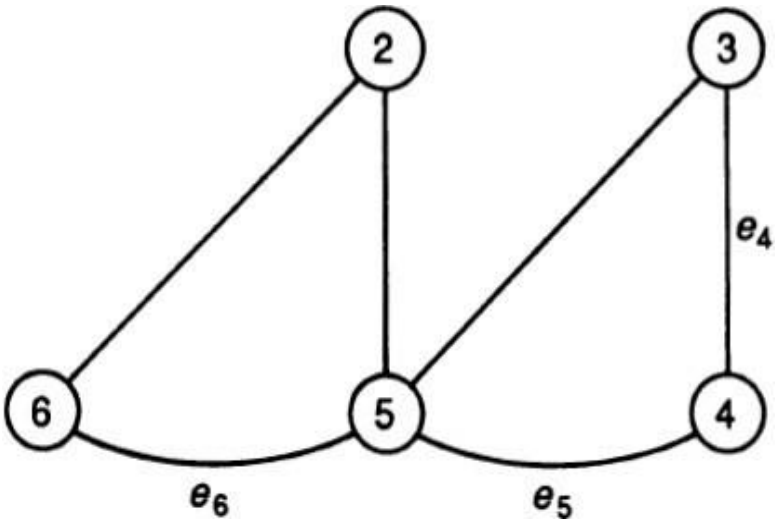
Gambar 6.1 Sirkuit euler



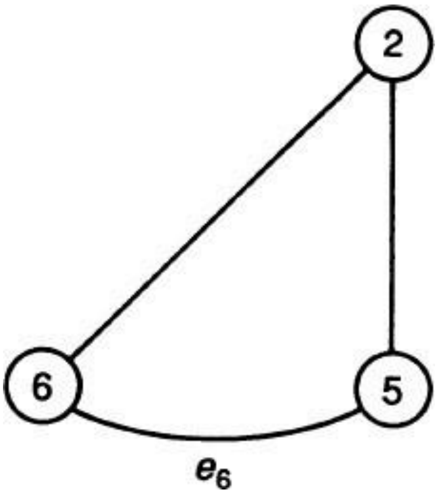
Gambar 6.2 Sirkuit euler

Untuk mengilustrasikan prosedur ini, mari kita coba membuat sirkuit Euler untuk graf G dari Gambar 6.2 di mana sisi-sisinya diberi label setiap kali ada banyak sisi. Mulai dari simpul 1, misalkan kita memiliki rangkaian C1, yang terdiri dari {1, 2}, {2, 3}, e3, e2, e1, dan {6, 1}. Menghapus semua tepi rangkaian ini dari G dan kemudian menghapus semua simpul dengan derajat nol, kita mendapatkan subgraf H1 seperti pada Gambar 6.3 dan kita melihat bahwa simpul 3 adalah umum untuk subgraf dan rangkaian C1. Dimulai dari simpul 3 pada subgraf ini, kita mendapatkan rangkaian C2 yang terdiri dari e4, e5 , dan {5, 3}. Kemudian kita sambung kedua rangkaian tersebut untuk mendapatkan rangkaian C3 yang terdiri dari {1, 2}, {2, 3}, semua rusuk C2, e3, e2, e1, dan {6, 1}. Sirkuit baru ini juga bukan Eulerian, meninggalkan kita dengan subgraf baru H2 seperti pada Gambar 6..4. Titik 2 adalah umum untuk subgraf ini dan C3 dan kami memiliki sirkuit C4 di H2 yang terdiri dari {2, 5}, e6 , dan {6, 2}. Akhirnya, kita sambung C4 dan C3 untuk mendapatkan rangkaian Euler dari G yang terdiri dari {1, 2}, {2, 5}, e6 , {6, 2}, {2, 3}, e4, e5, {5, 3 }, e3, e2, e1, dan {6, 1}.

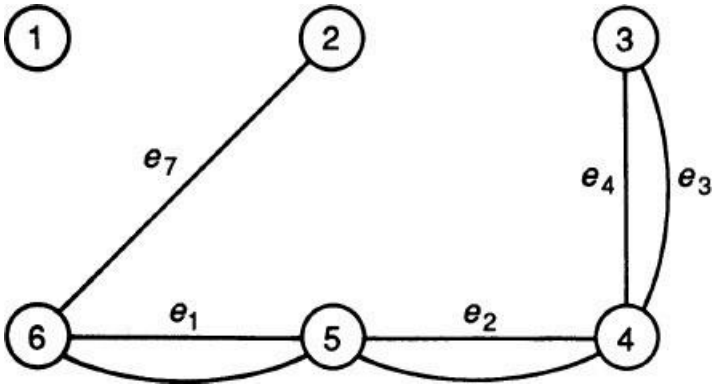
Menemukan sirkuit Euler dengan metode ini bisa jadi membosankan, terutama dalam grafik besar. Prosedur berikut, yang dikenal sebagai algoritma Fleury, tidak terlalu rumit: Mulai dari sembarang simpul dan hapus tepi segera setelah dilalui. Juga, jangan pernah menyeberangi jembatan jika Anda bisa membantunya. Jika kita dapat kembali ke titik awal setelah menghapus semua sisi, sirkuitnya adalah Euler dan kita menyimpulkan bahwa grafiknya juga Euler. Misalnya, pada Gambar 6.2 kita mulai dari 2 dan melintasi sepanjang {2, 3}, {3, 5}, {5, 2}, {2, 1}, dan {1, 6} dan berhenti di 6. Jika kita menghapus semua sisi yang dilalui, kita mendapatkan subgraf seperti pada Gambar 6.5, di mana kita mulai dari 6 tetapi kita tidak mengikuti e7 karena ini adalah jembatan. Jadi kita melintasi sepanjang e1, e2 , dan e3 , mencapai 3. Setelah tepi ini dihapus, e4 menjadi jembatan yang terpaksa kita lewati, dan dengan cara yang sama, kita menyeberangi jembatan e5, e6 dan akhirnya {6, 2}. Pada tahap ini kita memiliki sirkuit Euler.



Gambar 6.3 Simpul 3 adalah umum untuk subgraf dan rangkaian C1



Gambar 6.4 Subgraf baru H2



Gambar 6.5 Jika dihapus sisi yang dilalui akan mendapat subgraf baru

Catatan: Kehadiran loop di sebuah simpul sama sekali tidak mempengaruhi keberadaan sirkuit Euler. Misalkan G adalah sembarang graf terhubung dan misalkan G′ adalah subgraf yang diperoleh setelah menghapus semua loopnya. Maka G adalah Euler jika dan hanya jika G′ adalah Euler. Kecuali dinyatakan lain, kami berasumsi bahwa semua graf dan digraf dalam sisa bab ini adalah loopless.

Karakterisasi graf dengan jalur Euler sekarang dapat dengan mudah diperoleh sebagai berikut.

TEOREMA 6.1.2

Graf non-Euler terhubung G tanpa loop memiliki lintasan Euler jika dan hanya jika memiliki tepat dua simpul ganjil.

Bukti:

Jika G memiliki lintasan Euler dari u ke v, u dan v keduanya ganjil dan karena lintasan ini melewati setiap simpul dan melintasi setiap sisi satu kali, setiap simpul lainnya pasti genap. Sebaliknya, misalkan G terhubung dengan tepat dua simpul ganjil, u dan v. Sekarang u dan v bertetangga atau tidak. Dalam kasus sebelumnya biarkan e menjadi tepi di antara mereka. Hapus e untuk mendapatkan graf G′ (dengan paling banyak dua komponen) di mana setiap simpulnya genap. Jika G′ terhubung, dapatkan sirkuit Euler di dalamnya mulai dari u dan kemudian sambungkan tepi e ke sirkuit ini untuk mendapatkan jalur Euler antara u dan v. Jika G′ memiliki dua komponen, biarkan komponen yang berisi u menjadi G1 dan komponen yang mengandung v menjadi G2. Tentu saja, kedua komponen ini adalah Eulerian. Sekarang dapatkan sirkuit Euler dari u di komponen pertama dan sirkuit Euler dari v di komponen kedua. Kemudian jalur yang terdiri dari tepi sirkuit pertama, tepi (sebenarnya, jembatan) e, dan tepi sirkuit kedua merupakan jalur Euler antara u dan v. Akhirnya, jika u dan v tidak bertetangga di G, buat busur e yang menghubungkannya, menghasilkan grafik baru H, yang Eulerian. Dapatkan sirkuit Euler di H dari u di mana tepi terakhirnya adalah e. Jika kita menghapus e, kita memiliki jalur Euler di G dari u ke v. Ini melengkapi buktinya.

Kami sekarang menyatakan hasil analog dalam kasus digraf. Untuk bukti langsung teorema ini, pembaca dirujuk ke Behzad et al. (1979).

TEOREMA 6.1.3

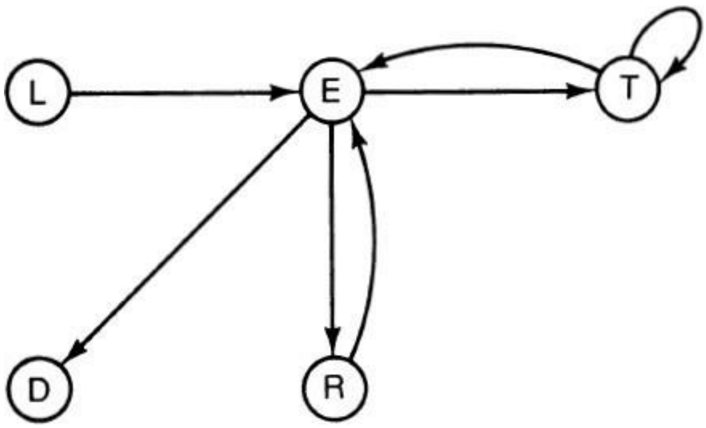
Digraf terhubung lemah memiliki sirkuit Euler berarah jika dan hanya jika derajat masuk setiap simpul sama dengan derajat keluarnya.

TEOREMA 6.1.4

Digraf terhubung lemah tanpa sirkuit Euler berarah memiliki lintasan Euler berarah jika dan hanya jika derajat masuk setiap simpul sama dengan derajat keluarnya kecuali untuk dua simpul u dan v sedemikian sehingga derajat keluar u sama dengan derajat masuk ditambah satu dan derajat masuk v sama outdegree ditambah satu.

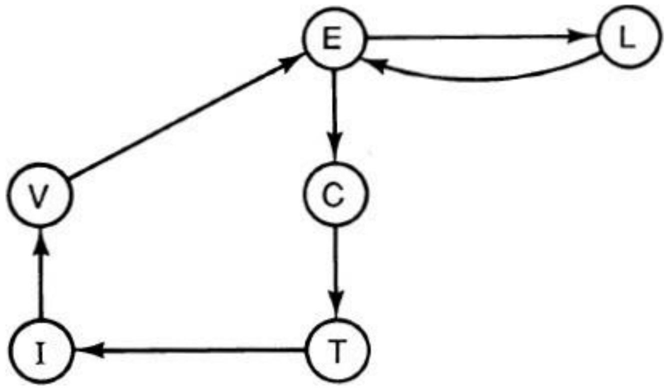
6.2 PENGKODEAN DAN DIGRAF DE BRUIJN

Ada beberapa aplikasi jalur dan sirkuit Euler yang menarik dan berguna di banyak bidang, seperti ilmu komputer, riset operasi, kriptografi, dan masalah transportasi, untuk beberapa nama. Kami membahas beberapa contoh di sini. Masalah tukang pos Cina adalah masalah optimasi jaringan di mana jaringan yang terhubung sewenang-wenang diperbesar menjadi jaringan Euler. Masalahnya dapat dinyatakan sebagai berikut: Seorang pembawa surat mulai dari kantor pos, mengirimkan surat ke setiap blok dengan ketukannya, dan kembali ke kantor pos. Jika kita mengambil setiap sudut jalan dalam rute sebagai simpul dan jalan antara dua sudut sebagai tepi, kita memiliki graf G sebagai model dari masalah ini. Jika G adalah Eulerian, pembawa surat harus melintasi setiap jalan tepat satu kali. Jika bukan Eulerian, dia harus mengulang beberapa sisi. Masalah optimasi yang khas dalam konteks ini adalah untuk menemukan jalan-jalan yang harus diulang sehingga total jarak yang dilalui adalah minimum. Ini pertama kali dibahas oleh ahli matematika Cina Kwan (1962) dan dikenal sebagai masalah tukang pos Cina. Pada bagian ini kita membahas penerapan graf Eulerian pada teori pengkodean.



Gambar 6.6 Dograf dengan lima titik dan tujuh titik

Setiap kata dengan m huruf yang n berbeda dapat dikaitkan dengan digraf G yang terhubung lemah dengan n simpul dan m – 1 busur sedemikian rupa sehingga kata tersebut mewakili jalur Euler berarah jika huruf pertama dan huruf terakhir berbeda dan Euler berarah rangkaian jika huruf pertama dan huruf terakhir sama. Misalnya, pada kata “LETTERED” kita memiliki m = 8 dan n = 5, dan huruf ini dapat diasosiasikan dengan jalur berarah dari titik L ke titik D pada digraf Gambar 6.6 dengan lima titik dan tujuh titik. busur diwakili oleh L- - -E- - -T- - -T- - -E- - -R- - -E- - -D, yang merupakan jalur Euler berarah. Demikian pula, kata "ELECTIVE" dapat dikaitkan dengan siklus Euler berarah E- - -L- - -E- --C- - -T- - -I- - -V- - -E pada digraf pada Gambar 6.7. Perhatikan bahwa meskipun sebuah kata mendefinisikan digraf secara unik, ada kemungkinan bahwa digraf yang sama dapat diasosiasikan dengan beberapa kata dengan panjang yang sama.



Gambar 6.7 Siklus euler berarah

Dalam kata apa pun dengan n huruf berbeda A1, A2, . . . , An, misalkan f(Ai) adalah frekuensi huruf Ai dalam kata tersebut. Maka jumlah frekuensi dari n huruf tersebut adalah m. Biarkan mij menunjukkan berapa kali Aj muncul segera setelah Ai, yang menunjukkan jumlah busur dari Ai ke Aj dalam digraf. Misalnya pada kata "MATEMATIKA", mAT = 2, mTA = 0, mTH = 1, dan seterusnya. Biarkan M = (mij) menjadi matriks n × n yang didefinisikan. Jumlah baris dari baris i pada M adalah derajat keluar dari simpul Ai, dan jumlah kolom dari kolom j adalah derajat masuk dari Aj.

Jadi, untuk setiap kata dengan n huruf yang berbeda, kita memiliki himpunan frekuensi n bilangan bulat positif dan matriks n × n yang elemen-elemennya adalah bilangan bulat nonnegatif. Misalnya, dalam kata "LETTERED" lima huruf yang berbeda adalah D, E, L, R, dan T, dengan frekuensi {1, 3, 1, 1, 2}, dan matriks 5 × 5 adalah:

$$\begin{array}{rc c}
 & \begin{array}{ccccc} \mathrm{D} & \mathrm{E} & \mathrm{L} & \mathrm{R} & \mathrm{T} \end{array} & \text{Row sum} \\
M = \begin{array}{c} \mathrm{D} \\ \mathrm{E} \\ \mathrm{L} \\ \mathrm{R} \\ \mathrm{T} \end{array} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} & \begin{array}{c} 0 \\ 3 \\ 1 \\ 1 \\ 2 \end{array} \\
\text{Column sum:} & \begin{array}{ccccc} 1 & 3 & 0 & 1 & 2 \end{array} &
\end{array}$$

Dalam kata "ELECTIVE" enam huruf yang berbeda adalah C, E, I, L, T, dan V, dengan himpunan frekuensi {1, 3, 1, 1, 1, 1}, dan matriks 6 × 6 adalah:

$$M = \begin{array}{c|cccccc|c} & C & E & I & L & T & V & \text{Row sum} \\ \hline C & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ E & 1 & 0 & 0 & 1 & 0 & 0 & 2 \\ I & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ L & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ T & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ V & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ \hline \text{Column sum:} & 1 & 2 & 1 & 1 & 1 & 1 & \end{array}$$

Kami sekarang membuat pernyataan yang mudah diverifikasi berikut ini:

1. Digraf kata apa pun terhubung lemah.
2. Jika huruf pertama dan huruf terakhir tidak sama, jumlah baris huruf pertama sama dengan jumlah kolom huruf pertama ditambah satu, jumlah baris huruf terakhir sama dengan jumlah kolom huruf terakhir dikurangi satu , dan untuk semua huruf lainnya jumlah baris dan jumlah kolom adalah sama.
3. Jika huruf pertama dan huruf terakhir sama, maka jumlah baris sama dengan jumlah kolom semua huruf dan jumlah baris huruf awal akan menjadi satu kurang dari frekuensinya.

Semua informasi dalam codeword terkandung dalam set frekuensi dan matriks M yang terkait dengan kata tersebut. Hutchinson dan Wilf (1975) mempelajari kata sandi dalam penyelidikan mereka terhadap molekul DNA dan RNA. Misalkan kita diberikan (a) himpunan n huruf A1, A2, . . . , An, (b) himpunan n bilangan bulat positif f1, f2, . . . , fn, dan (c) matriks n × n (mij) bilangan bulat tak negatif. Apakah ada kata di mana Ai muncul tepat lima kali dan Aj muncul tepat setelah Ai tepat lima kali? Jawabannya adalah “ya” jika kondisi yang sesuai dengan pernyataan (1)–(3) dipenuhi karena Teorema 6.1.3 dan 6.1.4. Dengan demikian kami memiliki hasil sebagai berikut.

TEOREMA 6.2.1

Misalkan M = (mij) adalah matriks n × n dengan komponen bilangan bulat nonnegatif dan misalkan Ai (i = 1, 2, . . , n) adalah himpunan dari n huruf yang berbeda sehingga Ai dikaitkan dengan baris i dan kolom i. Membiarkan:

ri = jumlah semua elemen baris i dari M
cj = jumlah semua elemen kolom j dari M

(a) Jika rj = cj + 1, rk = ck – 1, di mana j dan k berbeda dan jika ri = ci dalam semua kasus lainnya, terdapat sebuah kata yang dimulai dengan Aj dan diakhiri dengan Ak di mana frekuensi Aj adalah rj, frekuensi Ak adalah ck , dan frekuensi setiap huruf lainnya adalah ri, yang juga ci. Apalagi dalam kata tersebut, huruf Ap muncul tepat setelah Aq tepat mpq kali.
(b) Jika ri = ci untuk i = 1, 2, . . . , n, dan jika fi (i = 1, 2, . . . , n) adalah bilangan bulat tak negatif sehingga rj = fj – 1 dan ri = fi untuk setiap i selain j, terdapat kata yang diawali

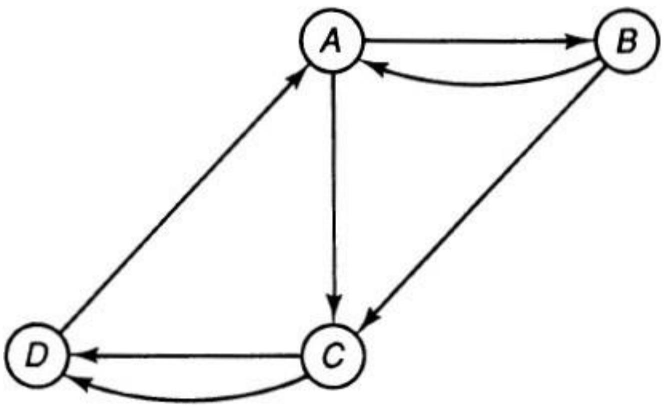
dengan Aj dan diakhiri dengan Aj di mana Ak muncul tepat kali fk dan Ap muncul setelah Aq tepat mpq kali.

Misalnya, misalkan huruf-huruf yang berbeda dalam sebuah kata adalah A, B, C, dan D dan matriksnya adalah:

$$\begin{array}{rc c}
 & \begin{array}{cccc} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} \end{array} & \text{Row sum} \\
M = \begin{array}{c} \mathrm{A} \\ \mathrm{B} \\ \mathrm{C} \\ \mathrm{D} \end{array} & \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 \end{bmatrix} & \begin{array}{c} 2 \\ 2 \\ 2 \\ 1 \end{array} \\
\text{Column sum:} & \begin{array}{cccc} 2 & 1 & 2 & 2 \end{array} &
\end{array}$$

Pertama kita buat digraf G dengan empat simpul A, B, C, dan D seperti pada Gambar 6.8. Kami mengamati:

1. Digraf terhubung dengan lemah.
2. (Jumlah baris untuk B) = (jumlah kolom untuk B) + 1.
3. (Jumlah baris untuk D) = (jumlah kolom untuk D) – 1.
4. Jumlah baris = jumlah kolom untuk semua huruf lainnya.



Gambar 6.8 Digraf dengan empat simpul

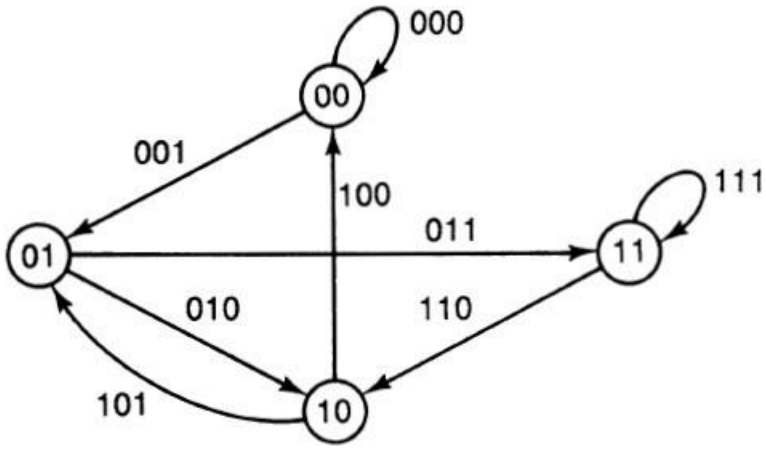
Jadi ada sebuah kata yang dimulai dengan B dan diakhiri dengan D yang dapat direpresentasikan dengan lintasan Euler dari B ke D pada digraf Gambar 6.8. Salah satu kata tersebut adalah BABCDACD. Ini adalah latihan yang mudah untuk menunjukkan bahwa jika himpunan frekuensi adalah {4, 3, 5, 2} dan matriksnya adalah

$$\begin{array}{rc}
 & \begin{array}{cccc} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} \end{array} \\
M = \begin{array}{c} \mathrm{A} \\ \mathrm{B} \\ \mathrm{C} \\ \mathrm{D} \end{array} & \begin{bmatrix} 1 & 0 & 2 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 2 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}
\end{array}$$

maka sebuah kata adalah CCBCBAADBDACAC.

Kami menyimpulkan bab ini dengan diskusi tentang digraf de Bruijn, aplikasi lain dari digraf Euler. Ada 2n–1 kata biner dengan panjang n – 1. Kami membuat digraf dengan simpul 2n–l sebagai berikut. Biarkan setiap kata dengan panjang n – 1 menjadi simpul. Dari setiap simpul berbentuk v = axa2 … an–1 tarik dua busur: satu ke a2a3 … an–1 0 dan yang lainnya ke a2a3 … an–1 1 untuk mewakili dua kata n-huruf v0 dan v1, masing-masing. Jadi busur 2n dari digraf yang dibangun mewakili himpunan kata-kata biner dengan panjang n. Digraf G(2, n) ini dikenal sebagai digraf de Bruijn, terhubung lemah dan bersifat Euler karena derajat masuk setiap simpul sama dengan derajat keluarnya. Digraf G(2, 3) seperti terlihat pada Gambar 6.9.

Lebih umum, untuk alfabet huruf p, G(p, n) adalah digraf de Bruijn dengan pn – 1 simpul dan pn busur sedemikian rupa sehingga derajat masuk dan derajat keluar setiap simpul keduanya p. Jadi G(p, n) adalah Euler. Sekarang perhatikan sirkuit Euler apa pun dalam digraf ini yang akan memuat semua busur pn secara berurutan. Bangun urutan huruf pertama dari semua kata ini. Mari kita nyatakan barisan ini dengan a1a2 … ar, di mana r = pn. Maka r kata berbeda dengan panjang n semuanya berbentuk aiai+1 … ai+n– 1, di mana operasi penjumlahan yang didefinisikan pada subskrip adalah modulo r. Misalnya, jika p = 2 dan n = 3, maka a9 adalah a1. Dalam digraf Gambar 6.9 rangkaian Euler berarah mulai dari 00 terdiri dari urutan delapan busur berikut: 000, 001, 011, 111, 110, 101, 010, 100. Huruf pertama dari busur ini membentuk kata 00011101 , sehingga a1 = 0, a2 = 0, a3 = 0, a4 = 1, a5 = 1, a6 = 1, a7 = 0, dan a8 = 1. Setiap kata yang terdiri dari tiga huruf sekarang berbentuk aiai + 1ai + 2. Jadi a7a8a9 = a7a8a1 = 010, dan seterusnya.



Gambar 6.9 Rangkaian euler berarah mulai dari urutan 00 delapan busur

Kita dapat mendefinisikan deret de Bruijn secara formal untuk dua bilangan bulat positif p dan n. Jika S adalah sembarang alfabet yang terdiri dari p huruf, maka barisan a1a2 • • • ar dari r (r = pn) huruf disebut barisan de Bruijn, dilambangkan dengan B(p, n) , jika setiap kata dengan panjang n dari S dapat diwujudkan sebagai aiai+l • • • ai + n – 1 (i = 1, 2, . . , r), dimana operasi penjumlahan pada subscriptnya adalah modulo r.

Kami sekarang siap untuk meringkas pengamatan kami sebagai teorema.

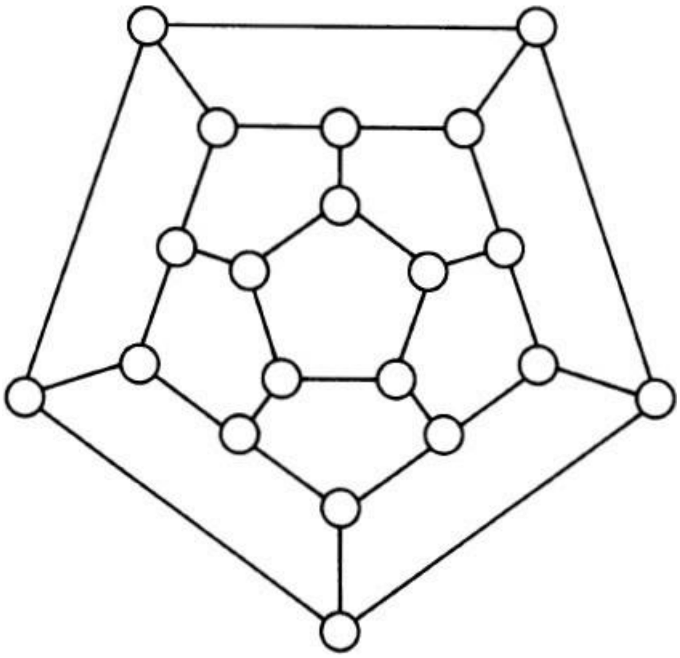
TEOREMA 6.2.2

Untuk setiap pasangan bilangan bulat positif terdapat barisan de Bruijn.

Ini pertama kali dibuktikan oleh de Bruijn (1946) untuk p = 2 dan kemudian digeneralisasikan untuk p sewenang-wenang oleh Good (1946). Urutan ini sangat berguna dalam teori pengkodean. Diagram keadaan dari register geser umpan balik (FSR) adalah subgraf dari digraf de Bruijn tertentu. FSR memiliki berbagai aplikasi dalam komunikasi, kriptografi, dan ilmu komputer, terutama karena sifat keacakan dari urutan yang mereka hasilkan. Secara singkat, jika K adalah medan (orde q) , dan jika f: Kn → K, maka FSR tingkat-n pada K mentransformasikan vektor [x0 x1 • • • xn – 1] menjadi vektor [x1 x2 • •• xn] , di mana xn = f(x0, x1 , . . , xn – 1).

6.3 JALUR HAMILTONIAN DAN SIKLUS HAMILTON

Lintasan antara dua simpul dalam graf adalah lintasan Hamilton jika melewati setiap simpul tepat satu kali. Lintasan tertutup yang melalui setiap simpul tepat satu kali dan di mana semua sisinya berbeda adalah siklus Hamilton. Suatu graf adalah graf Hamilton jika memiliki siklus Hamilton. Dalam digraf, suatu lintasan berarah dari suatu simpul ke simpul lain adalah lintasan Hamiltonian berarah jika lintasan itu melalui setiap simpul tepat satu kali. Lintasan Hamilton berarah tertutup adalah siklus Hamilton berarah. Kata sifat "Hamiltonian" adalah untuk menghormati matematikawan Irlandia terkenal Sir William Hamilton (1805– 1865), yang menyelidiki keberadaan solusi untuk permainan yang disebut "seluruh dunia," di mana pemain diminta untuk menemukan rute sepanjang tepi dodecahedron (polihedron biasa dengan 20 simpul, 30 tepi, dan 12 wajah) mengunjungi setiap simpul tepat satu kali dan kembali ke simpul awal. Sekarang sebuah dodecahedron dapat direpresentasikan sebagai graf G pada bidang (lihat Gambar 6.10) dengan 20 simpul dan 30 tepi. Dengan demikian, permainan memiliki solusi jika dan hanya jika G adalah graf Hamilton.



Gambar 6.10 Graf G dengan 20 simpul dan 30 tepi

Meskipun masalah penentuan keberadaan siklus Hamilton tampaknya serupa dengan masalah penentuan keberadaan sirkuit Euler, sama sekali tidak mudah untuk mengatakan apakah suatu graf yang diberikan adalah Hamiltonian secara umum. Berbeda dengan kondisi perlu dan cukup yang sangat rapi yang diperoleh Euler untuk keberadaan sirkuit Euler, grafik Hamilton tampaknya menentang karakterisasi. Dalam banyak kasus, setiap grafik harus dipertimbangkan secara individual karena tidak ada kondisi perlu yang mudah diverifikasi yang diketahui secara umum. Tentu saja, graf lengkap adalah Hamiltonian. Dengan kata lain, sebuah graf dengan n simpul adalah Hamiltonian jika derajat setiap simpul adalah n – 1. Semakin besar derajat setiap simpul, semakin besar kemungkinan bahwa graf tersebut adalah Hamiltonian. Jadi pertanyaannya adalah: Apakah terdapat bilangan bulat positif k (k < n – 1) sehingga graf tersebut adalah Hamiltonian jika derajat setiap simpul paling sedikit k? Jawabannya adalah ya, sebagaimana dibuktikan oleh Dirac (1952), yang teoremanya juga dapat diperoleh sebagai konsekuensi dari teorema Bijih (1963).

TEOREMA 6.3.1

Graf sederhana dengan n simpul (di mana n paling sedikit 3) adalah Hamiltonian jika jumlah derajat setiap pasangan simpul yang tidak berdekatan paling sedikit n.

Bukti:

Misalkan graf G dengan n simpul bukan Hamiltonian. Jadi itu adalah subgraf dari graf lengkap Kn dengan sisi yang lebih sedikit. Sekarang terus menambahkan tepi ke G dengan menggabungkan simpul yang tidak berdekatan sampai kita memperoleh graf non-Hamilton H sedemikian rupa sehingga penambahan satu sisi lagi ke H akan menjadikannya Hamiltonian. Misalkan x dan y adalah sembarang pasangan simpul yang tidak berdekatan di H. Jadi mereka juga tidak berdekatan di G. Jadi (deg x + deg y) paling sedikit n di H. Karena penambahan {x, y} sebagai edge pada H akan menjadikannya Hamiltonian, ada lintasan Hamiltonian di H antara x dan y. Jika kita menulis x = v1 dan y = vn, maka jalur ini dapat ditulis sebagai v1- - -v2- - -v3 --- vi – 1 - - - vi- - -vi + 1. . .vn – 1- - -vn. Perhatikan bahwa jika v1 dan vi bertetangga di H, maka vn dan vi – 1 tidak dapat bertetangga karena jika keduanya bertetangga, kita akan memiliki siklus Hamiltonian berikut di H: vn- - -vi – 1, . . . v1 - - -vi . . .vn, yang merupakan kontradiksi. Jadi jika v1 memiliki r simpul yang berdekatan dari himpunan {v2, v3, . . . , vn}, setidaknya r simpul dari himpunan {v1, v2, . . . , vn – 1} tidak boleh berdekatan dengan vn. Dalam kasus tersebut, deg v1 = r dan deg vn (n – 1) – r dan akibatnya, deg v1 + deg vn (n – 1) < n, yang bertentangan dengan hipotesis.

COOLLARY(Teorema Dirac)

Graf sederhana dengan n simpul adalah Hamiltonian jika derajat setiap simpul paling sedikit n/2.

Catatan: Kebalikan dari teorema Bijih tidak benar. Misalnya, perhatikan grafik poligon dengan enam sisi. Kondisi cukup untuk keberadaan Jalur Hamilton dalam suatu graf adalah seperti pada hasil berikut.

TEOREMA 6.3.2

Graf sederhana dengan n simpul memiliki lintasan Hamilton jika jumlah derajat setiap pasangan simpul yang tidak berdekatan paling sedikit (n – 1).

Bukti:

Ini adalah latihan.

Akibat Wajar

Graf sederhana dengan n simpul memiliki lintasan Hamilton jika derajat setiap simpul paling sedikit (n – l)/2.

Sama seperti dalam kasus graf, tidak ada karakterisasi yang diketahui dari digraf Hamiltonian. Bahkan, situasinya menjadi lebih kompleks. Kami menyatakan di sini beberapa kondisi yang cukup untuk keberadaan siklus dan jalur Hamilton langsung dalam digraf sederhana yang kurang lebih mirip dengan hasil dalam kasus graf. Lihat Behzad dkk. (1979) untuk bukti.

TEOREMA 6.3.3

(a) Digraf terhubung kuat dengan n simpul adalah digraf Hamilton jika (derajat u + derajat v) paling sedikit 2n – 1 untuk setiap pasangan simpul u dan v sedemikian sehingga tidak ada busur dari u ke v dan dari v ke kamu

(b) Digraf dengan n simpul adalah Hamiltonian jika (derajat keluar u + derajat dalam v) paling sedikit n untuk setiap pasangan simpul u dan v sehingga tidak ada busur dari u ke v.

(c) Digraf terhubung kuat dengan n simpul adalah Hamiltonian jika (derajat keluar v + derajat masuk v) paling sedikit n untuk setiap simpul v.

(d) Suatu digraf dengan n simpul adalah Hamiltonian jika derajat keluar dan derajat masuk masing-masing simpul paling sedikit n/2.

TEOREMA 6.3.4

(a) Jika (derajat u + derajat v) paling sedikit 2n – 3 untuk setiap pasang simpul u dan v sedemikian rupa sehingga tidak ada busur dari satu ke yang lain dalam digraf G, maka G memiliki lintasan Hamiltonian berarah.

(b) Jika (derajat keluar u + derajat masuk v) paling sedikit (n – 1) untuk setiap pasangan simpul sedemikian rupa sehingga tidak ada busur dari u ke v dalam digraf G, maka G memiliki lintasan Hamiltonian berarah.

(c) Jika (derajat keluar v + derajat masuk v) paling sedikit (n – 1) untuk setiap simpul v dalam digraf dengan n simpul, digraf tersebut memiliki lintasan Hamiltonian berarah.

(d) Jika derajat keluar dan derajat masuk masing-masing simpul paling sedikit (n – l)/2 dalam digraf G, maka G memiliki lintasan Hamiltonian berarah.

Graf Terhubung Hamiltonian

Suatu graf dikatakan terhubung Hamiltonian jika terdapat lintasan Hamilton di antara setiap pasangan simpul di dalamnya. Jelas setiap graf terhubung Hamilton dengan tiga atau lebih simpul pasti merupakan graf Hamilton. Kebalikannya tidak benar: Dalam graf Hamilton G = (V, E) di mana V = {1, 2, 3, 4} dan E adalah himpunan {{1, 2}, {2, 3}, {3, 4}, {4, 1}} tidak ada lintasan Hamiltonian antara simpul 2 dan simpul 4. Hasil berikut karena Bijih (1963) sejajar dengan hasil Teorema 6.3.1.

TEOREMA 6.3.5

Jika G adalah graf sederhana dengan n simpul (n paling sedikit 3) sedemikian sehingga untuk semua simpul berbeda i dan j, (derajat i) + (derajat j) melebihi n, maka G terhubung Hamiltonian. Untuk detail lebih lanjut tentang graf terhubung Hamilton, lihat makalah oleh Chartrand et al. (1969) dan Jilat (1970).

6.4 APLIKASI SIKLUS HAMILTONIAN

Lintasan dan siklus Hamilton memiliki beberapa aplikasi yang berguna dan menarik. Kami membahas beberapa di antaranya di sini.

Contoh 6.4.1 (The Travelling Salesman Problem)

Pada Contoh 5.1.3 kami memperkenalkan travelling salesman problem (TSP), di mana seorang salesman harus membuat rencana perjalanan mengunjungi setiap kota dalam tur tepat satu kali dan kembali ke titik awal. Tur semacam itu adalah siklus Hamilton. Dengan asumsi bahwa siklus seperti itu ada, masalah optimasi kemudian adalah menemukan tur yang biaya totalnya (atau dalam hal ini, jarak total) adalah minimum. TSP adalah salah satu masalah paling terkenal dari kelas yang mudah dinyatakan tetapi sangat sulit untuk dipecahkan. Secara umum, tidak ada prosedur efisien yang diketahui untuk menemukan solusi dari masalah tersebut. Jika kita mengadopsi metode enumerasi lengkap di mana kita mendaftar semua $(n - 1)!$ siklus Hamiltonian terarah (dalam kasus terburuk) adalah digraf dengan n simpul, dan hitung biaya untuk setiap siklus dengan melakukan n penambahan, kita akan melakukan $n(n - 1)!$ tambahan. Jika $n = 20$ dan komputer dapat melakukan 1 juta penambahan per detik, metode ini akan memakan waktu sekitar 75.000 tahun. Lihat Held dan Karp (1970) untuk diskusi tentang pertimbangan kompleksitas yang terlibat dalam masalah ini.

Contoh 6.4.2 (Penjadwalan)

Pertimbangkan sebuah toko mesin dengan n mesin yang berbeda. Pekerjaan harus dijalankan melalui semua mesin ini tetapi tidak dalam urutan tertentu. Biarkan setiap mesin mewakili simpul digraf. Gambarlah busur dari setiap simpul ke setiap simpul lainnya. Maka setiap lintasan Hamiltonian berarah dalam digraf adalah suatu jadwal. Jika cij adalah waktu setup yang diperlukan setiap kali pekerjaan berpindah dari mesin i ke mesin j, masalah optimasi adalah menemukan jadwal yang membutuhkan waktu paling sedikit.

Contoh 6.4.3

Dalam Contoh 5.1.4 kami mendefinisikan turnamen sebagai digraf sederhana di mana untuk setiap pasangan simpul v dan w ada busur dari v ke w atau dari w ke v tetapi tidak keduanya. Dengan kata lain, grafik yang mendasari turnamen selesai dan (v, w) adalah busur di turnamen jika dan hanya jika (w, v) bukan busur. Jika simpul menunjukkan pemain yang berbeda, keberadaan busur dari v ke w menunjukkan bahwa v mengalahkan w dalam permainan. Pertanyaan-pertanyaan berikut muncul: (1) Apakah mungkin untuk membuat peringkat semua pemain sebagai urutan u1, u2, . . . , un sedemikian rupa sehingga ui, mengalahkan ui+1 (i = 1, 2, . . , , n – 1)? Dengan kata lain, apakah digraf memiliki lintasan Hamiltonian berarah? (2) Jika jalur seperti itu ada, apakah itu unik? (3) Apakah ada syarat perlu dan cukup yang harus dipenuhi oleh digraf agar lintasannya unik, yang menyiratkan bahwa peringkatnya juga unik? Jawabannya adalah: (1) ya, (2) tidak, dan (3) ya.

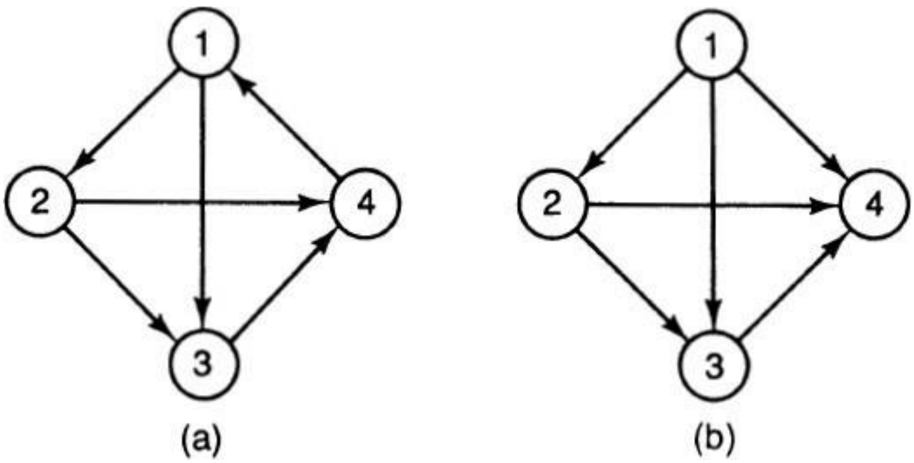
Sebelum kita membenarkan pernyataan ini, mari kita perhatikan dua turnamen (a) dan (b), seperti pada grafik Gambar 6.11. Kami melihat bahwa kedua turnamen memiliki jalur Hamilton. Dalam (a) kita memiliki empat jalur Hamilton yang berbeda, sedangkan pada (b) kita hanya memiliki satu. Apa yang membuat keduanya berbeda? Berikut adalah definisi untuk mengatasi hal tersebut. Sebuah turnamen dikatakan transitif jika setiap kali u mengalahkan v dan v mengalahkan w, maka u mengalahkan w. Perhatikan bahwa turnamen transitif jika dan hanya jika tidak memiliki siklus terarah dengan tiga busur. Meskipun "transitivitas" tampak normal di sebagian besar situasi dalam kehidupan, di dunia nyata sebagian besar turnamen tidak bersifat transitif. Mungkin ini salah satu alasan mereka menarik. Dalam ilustrasi kita, (b) transitif tetapi (a) tidak. Teorema pertama kami di turnamen adalah karena Redei (1934).

TEOREMA 6.4.1

Setiap turnamen G memiliki jalur Hamiltonian terarah.

Bukti:

Ini adalah konsekuensi langsung dari Teorema 6.3.4(c). Namun, pembuktian independen menggunakan induksi pada jumlah n simpul adalah sepanjang baris berikut. Teorema ini benar jika n = 2. Anggaplah benar untuk n. Sekarang pertimbangkan setiap turnamen G′ dengan (n + 1) simpul. Misalkan v adalah sembarang simpul sembarang dari G′. Sekarang perhatikan subgraf G dari G′ yang diperoleh dari G′ dengan menghapus v dan semua busur dari v dan ke v. Jelas, G adalah turnamen dengan n simpul sehingga memiliki jalur Hamiltonian berarah v1 - - > v2 . . . > vi . . . > vn. Jika ada busur di G′ dari v ke v1 atau dari vn ke v, maka G′ memiliki jalur Hamiltonian berarah dan kita selesai. Jika tidak, biarkan i menjadi bilangan bulat terbesar sehingga tidak ada busur dari v ke vi. Jadi ada busur dari vi ke v. Sekarang pilihan i kita sedemikian rupa sehingga tidak ada busur dari vi+1 ke v yang menyiratkan bahwa ada busur dalam arah yang berlawanan dan akibatnya kita memiliki jalur Hamiltonian berarah di G′ sebagai berikut: v1. . . > vi- - - > v- - - > vi+1. . . menunjukkan bahwa hasilnya benar untuk (n + 1).



Gambar 6.11 dua turnamen (a) dan (b)

Akibat Wajar

Turnamen transitif memiliki jalur Hamiltonian terarah yang unik.

Bukti:

Jika P dan P′ adalah dua jalur Hamiltonian berarah yang berbeda, ada sepasang simpul x dan y sedemikian rupa sehingga ada lintasan dari x ke y di P dan lintasan dari y ke x di P′. Jadi, dengan transitivitas, ada busur, dari x ke y di P dan busur dari y ke x di P′. Ini adalah kontradiksi. Jadi P = P′.

Kami menyimpulkan diskusi kami tentang turnamen dengan teorema berikut tentang keberadaan jalur Hamiltonian terarah yang unik dalam turnamen. Lihat Roberts (1976) untuk bukti.

TEOREMA 6.4.2

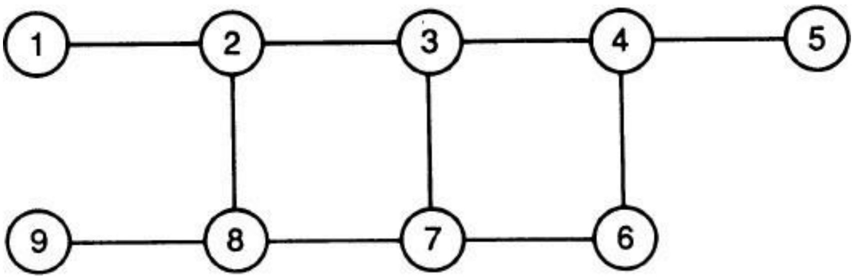
Dalam turnamen G properti berikut ini setara:

(a) G memiliki lintasan Hamiltonian terarah yang unik.
(b) G tidak memiliki siklus berarah dengan panjang 3.
(c) G adalah asiklik.
(d) G transitif.

6.5 PEWARNAAN VERTEKS DAN PLANARITAS GRAFIK

Suatu graf dikatakan berwarna jika setiap simpul diberi warna sedemikian rupa sehingga tidak ada dua simpul bertetangga yang memiliki warna yang sama. Jika penetapan warna seperti itu dimungkinkan dengan menggunakan paling banyak k warna, grafiknya adalah k-warna. Nilai k terkecil sehingga graf G dapat diwarnai dengan k adalah bilangan kromatik G.

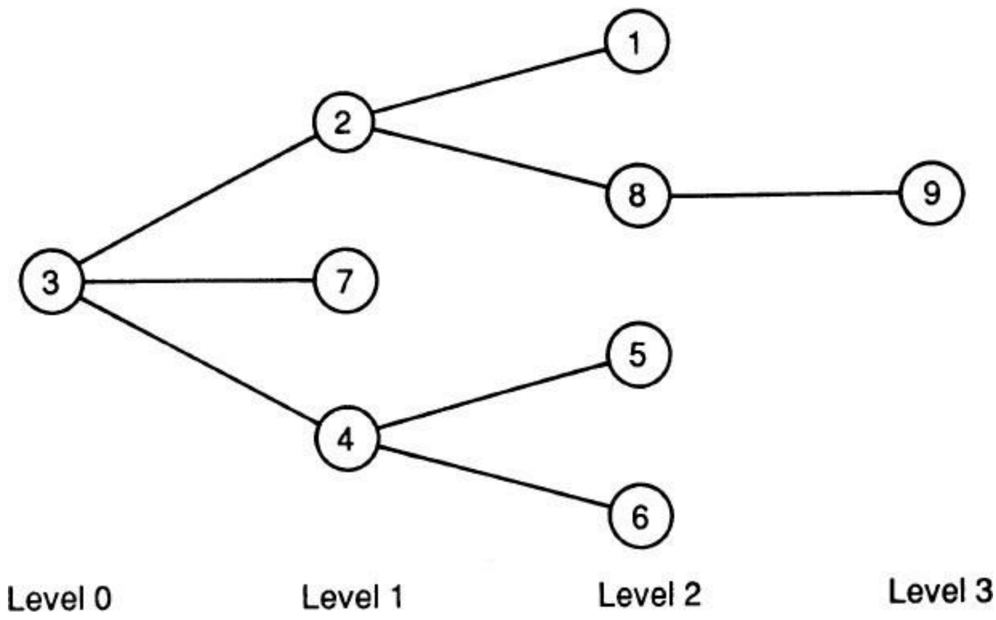
Bilangan kromatik suatu graf adalah 1 jika dan hanya jika graf tersebut tidak memiliki rusuk. Bilangan kromatik graf lengkap dengan n simpul tentu saja n, dan bilangan kromatik graf bipartit adalah 2. Secara khusus, bilangan kromatik pohon adalah 2. Setiap siklus dengan p simpul dapat diwarnai 2 jika dan hanya jika p genap. Akibatnya, jika graf G memiliki siklus ganjil (yaitu, siklus dengan jumlah simpul ganjil), G tidak dapat diwarnai 2 warna. Di sisi lain, jika tidak ada siklus ganjil dalam graf G, graf tersebut memiliki 2 warna. Ini jelas jika G adalah pohon karena pohon adalah asiklik. Lebih umum, asumsikan bahwa G adalah graf terhubung tanpa siklus ganjil. Mulai dari sembarang simpul v dan terapkan prosedur breadth first search (BFS) untuk mendapatkan pohon BFS sebagai berikut: v berada pada level 0. Semua simpul yang berdekatan dengan v berada pada level 1. Kami mempartisi simpul dari graf menjadi himpunan simpul di berbagai tingkatan. Misalkan vi1, vi2, . . . , vir menjadi simpul pada tingkat i. Pertimbangkan semua simpul yang berdekatan dengan vi2 yang tidak berada pada level 0, 1, 2, . . . ,saya. Letakkan simpul-simpul ini di level baru (i + 1). Kemudian pertimbangkan semua simpul yang bertetangga dengan vi2 tetapi tidak pada level 0, 1, 2, . . . , saya, (i + 1). Sertakan simpul-simpul ini di level (i + 1). Lanjutkan proses ini sampai semua simpul diperiksa. Kami sekarang menetapkan dua warna ke simpul: satu warna untuk semua simpul di tingkat ganjil dan warna lain untuk semua simpul di tingkat genap. Pada Gambar 6.12 kita memiliki graf tanpa siklus ganjil dan pohon BFS mulai dari simpul 3 seperti pada Gambar 6.13.



Gambar 6.12 Graf tanpa siklus ganjil

Dalam prosedur BFS dalam kasus terburuk kita harus memeriksa semua n simpul dan semua m tepi. Jadi kompleksitas kasus terburuk adalah n + m. Karena m paling banyak n(n – l)/2, kompleksitasnya adalah n(n + 1)/2.

Dari latihan sebelumnya kita tahu bahwa graf adalah bipartit jika dan hanya jika tidak memiliki siklus ganjil. Jadi kita memiliki teorema berikut untuk mengkarakterisasi 2-warna dari grafik.



Gambar 6.13 Pohon BFS mulai dari 3 simpul

TEOREMA 6.5.1

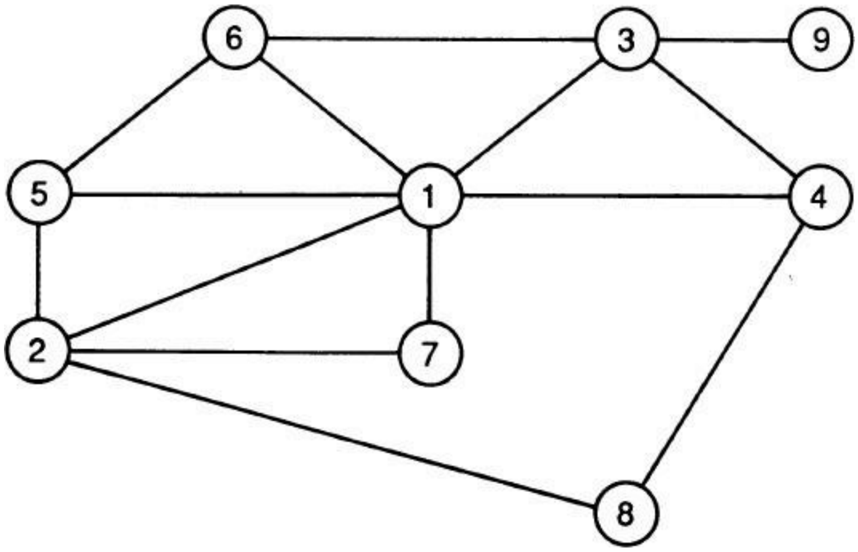
Dalam graf terhubung G, berikut ini adalah setara:

(a) G adalah bipartit.
(b) G adalah 2-warna.
(c) G tidak memiliki siklus ganjil.

Tidak ada karakterisasi yang diketahui untuk k-warna dari grafik ketika k > 2. Secara umum, ini adalah masalah yang sulit untuk menghitung bilangan kromatik dari grafik arbitrer. Tidak ada algoritma yang selalu memberikan pola pewarnaan dengan menggunakan warna sesedikit mungkin. Namun, ada algoritme untuk mewarnai grafik tertentu yang "mendekati" pewarnaan terbaik dalam arti bahwa terkadang grafik tersebut menggunakan lebih banyak warna daripada yang mutlak diperlukan. Berikut adalah algoritma, yang dikenal sebagai algoritma terbesar pertama, karena memberikan warna pada simpul dengan derajat terbesar terlebih dahulu. Pertama kita mengurutkan simpul menurut derajat yang tidak bertambah. Gunakan warna pertama untuk mewarnai simpul pertama dan kemudian warnai, secara berurutan, setiap simpul yang tidak berdekatan dengan simpul yang sebelumnya diwarnai dengan warna yang sama. Ulangi proses ini menggunakan warna kedua untuk urutan simpul yang tidak berwarna. Lanjutkan proses ini sampai semua simpul diwarnai.

Mari kita gunakan algoritma terbesar pertama untuk mendapatkan pewarnaan graf pada Gambar 6.14. Sembilan simpul diberi label sebagai i = 1, 2, 3, . . . , 9, di mana derajat i lebih besar dari atau sama dengan derajat (i + 1). Pertama berikan warna 1 ke simpul 1. Sekarang simpul 8 dan 9 tidak bertetangga untuk 1. Jadi tetapkan warna 1 ke simpul 8. Kita lihat bahwa simpul 9 tidak bertetangga dengan simpul 8. Jadi tetapkan warna 1 ke simpul 9 juga. Jadi simpul 1, 8, 9 diberi warna 1. Simpul yang tersisa adalah 2, 3, 4, 5, 6, 7. Beri warna 2 ke simpul 2. Simpul yang tidak berwarna yang tidak berdekatan dengan simpul 2 adalah 3, 4, 6. Beri warna 2 pada simpul 3. Kita perhatikan bahwa simpul 3 dan 4 bertetangga. Jadi simpul

4 tetap tidak berwarna. Demikian pula, simpul 6 tetap tidak berwarna. Jadi simpul 2 dan 3 diwarnai dengan warna 2.



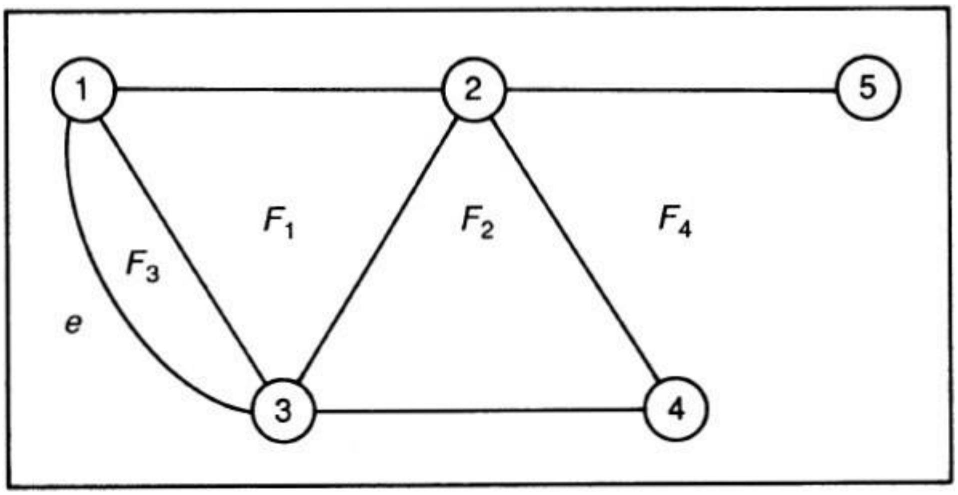
Gambar 6.14 Graf dengan Sembilan simpul

Simpul tak berwarna yang tersisa adalah 4, 5, 6, 7. Tetapkan warna 3 ke simpul 4 dan kemudian ke simpul 5 dan simpul 7. Akhirnya, kita tersisa dengan simpul 6, yang memberi warna 4. Jadi grafik Gambar 5.5. 3 tidak membutuhkan lebih dari empat warna. Tapi tiga warna akan melakukan pekerjaan itu. Tetapkan warna merah ke 1, warna biru ke 3, 5, 7, 8, dan warna hijau ke 2, 4, 6, 9.

Ide mewarnai graf muncul secara alami dalam banyak masalah penjadwalan. Misalkan departemen ilmu komputer di sebuah universitas telah memutuskan untuk menawarkan sejumlah program pascasarjana dan ada sejumlah periode waktu tertentu selama kursus ini dapat ditawarkan. Dalam menjadwalkan mata kuliah ini departemen harus menghindari konflik. Jika seorang mahasiswa pascasarjana tertarik untuk mengambil dua mata kuliah ini, mata kuliah tersebut harus dijadwalkan pada waktu yang berbeda. Kami membangun model grafik dari masalah penjadwalan ini sebagai berikut: Biarkan setiap simpul mewakili jalur. Gabungkan dua simpul dengan tepi jika kursus yang sesuai dengan dua simpul ini tidak dapat ditawarkan pada saat yang sama. Jika grafik yang dihasilkan adalah k-warna, kursus dapat dijadwalkan dengan k periode atau lebih.

Planaritas Grafik

Suatu graf disebut graf planar jika dapat digambarkan sedemikian rupa sehingga tidak ada dua sisi yang berpotongan kecuali pada satu titik. Graf planar yang digambar pada bidang sehingga tidak ada dua sisi yang berpotongan adalah graf bidang. Daerah dua dimensi yang didefinisikan oleh tepi dalam grafik bidang adalah wajah dan simpul dan berbagai tepi mendefinisikan batas-batas wajah ini. Pada graf bidang Gambar 6.15 terdapat lima simpul, tujuh sisi, dan empat wajah. Dalam grafik bidang ini, kita melihat bahwa F1, F2 , dan F3 adalah permukaan dalam, dan daerah tak terbatas F4 adalah permukaan luar. Batas F1 adalah lingkaran 1- - - - -2- - - - -3- - -- -1 dan batas F4 adalah 1- - - - -2- - - - -5- - - - - 2- - - - -4- - - - -3 1, di mana tepi terakhirnya adalah e.



Gambar 6.15 Graf dengan lima simpul

Hasil klasik (dibuktikan pada 1750) yang menghubungkan jumlah simpul, tepi, dan wajah adalah teorema Euler berikut.

TEOREMA 5.5.2

Jika suatu graf bidang terhubung memiliki n titik dan m sisi, maka jumlah sisinya adalah p, dimana n – m + p = 2.

Bukti:

Kami menggunakan induksi pada m. Ketika m = 0, kita memiliki n = 1 dan p = 1 dan hasilnya benar. Misalkan hasilnya benar ketika m = k – 1. Pertimbangkan sembarang graf bidang G dengan n simpul, k sisi dan p wajah. Kami ingin menunjukkan n – k + p = 2. Hal ini tentu benar jika graf adalah pohon. Jika bukan pohon, misalkan e adalah sembarang sisi dari sebuah siklus. Jika kita menghapus e, kita masih memiliki graf bidang terhubung dengan n simpul, (k – 1) tepi, dan (p – 1) wajah. Perhatikan bahwa jika kita menghapus sebuah sisi dari sebuah siklus, dua menyatu menjadi satu. Dengan hipotesis induksi kami, n – (k – 1) + (p – 1) = 2, jadi n – k + p = 2, seperti yang ingin kami buktikan.

Kami sekarang menetapkan hasil lain yang bermanfaat, yang merupakan konsekuensi langsung dari apa yang telah kami buktikan.

TEOREMA 5.5.3

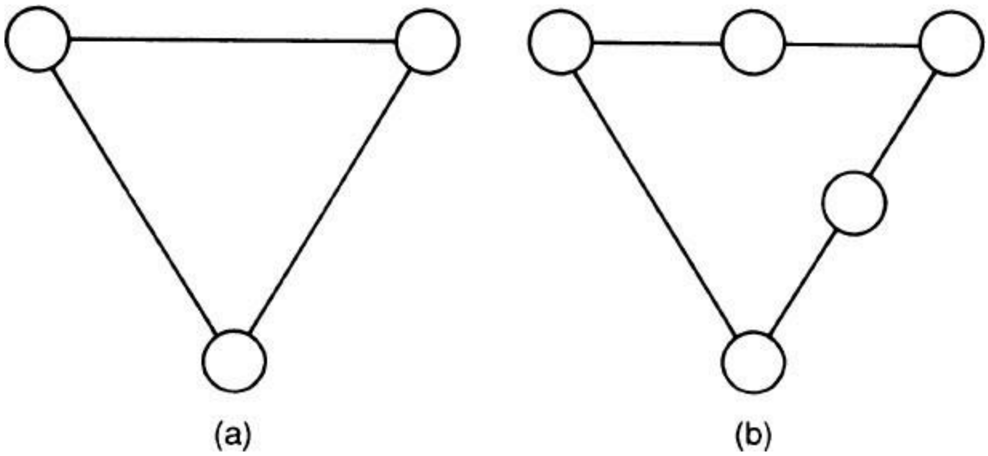
Graf planar terhubung sederhana dengan n simpul (n minimal 3) memiliki paling banyak (3n – 6) rusuk.

Bukti:

Jika n adalah 3, maka jumlah rusuknya paling banyak tiga. Biarkan n lebih besar dari atau sama dengan 3. Kami menggambar grafik bidang dengan wajah F1 F2, . . . , Fp. Biarkan ri menjadi jumlah tepi yang mendefinisikan wajah Fi. Maka ri setidaknya tiga untuk setiap i. Jadi 3p (r1 + r2 + . . . + rp) . Sekarang dalam menghitung jumlah total tepi dalam batas, setiap tepi dihitung paling banyak dua kali. Jadi ruas kanan pertidaksamaan di atas paling banyak 2m, di mana m adalah jumlah rusuk pada grafik. Oleh karena itu 3p paling banyak 2m. Tetapi dengan Teorema 6.5.2 kita tahu bahwa p = 2 – n + m. Hasilnya berikut.

Kami menggunakan teorema ini untuk menetapkan nonplanaritas dari beberapa grafik terkenal. Jika graf sederhana dengan n simpul memiliki lebih dari (3n – 6) rusuk, maka graf tersebut tidak planar. Graf lengkap K5 memiliki lima simpul dan 10 rusuk, sehingga tidak planar. Kita dapat menetapkan nonplanaritas dari graf bipartit lengkap K3, 3 dengan kontradiksi. Untuk graf ini kita mengetahui bahwa n = 6 dan m = 9. Jadi jika graf tersebut planar, harus tepat memiliki lima wajah sesuai dengan Teorema 6.5.2. Ingatlah bahwa graf bipartit tidak memiliki siklus ganjil. Jadi harus ada setidaknya 20 tepi untuk menentukan batas-batas wajah ini. Setiap tepi dihitung paling banyak dua kali. Jadi grafik harus memiliki setidaknya 10 tepi. Tetapi hanya ada 9. Jadi setiap graf yang memiliki K5 atau K3,3 sebagai subgrafnya adalah nonplanar. Ternyata setiap graf nonplanar mengandung salah satu dari keduanya sebagai subgraf dalam arti tertentu yang sekarang kita perinci.

Dua graf dikatakan homeomorfik (atau identik satu sama lain dalam simpul berderajat 2) jika keduanya dapat diperoleh dari graf G yang sama dengan memasukkan simpul baru berderajat 2 pada rusuknya. Misalnya, dua graf (a) dan (b) pada Gambar 6.16 bersifat homeomorfik. Perhatikan bahwa penyisipan atau penghapusan simpul pada tepi tidak mempengaruhi pertimbangan planaritas. Kami sekarang menyatakan teorema terkenal Kuratowski berikut (dibuktikan pada tahun 1930), yang memberikan kondisi yang diperlukan dan cukup untuk grafik menjadi planar. Lihat Bondy dan Murty (1976) untuk bukti.



Gambar 6.16 Graf dengan sifat homeomorfik

TEOREMA 6.5.4

Suatu graf planar jika dan hanya jika tidak mengandung subgraf yang homeomorfik terhadap K5 atau K3,3.

Terakhir, catatan tentang pewarnaan peta. Ketika kita mewarnai negara yang berbeda dalam peta geografis, dua negara dengan perbatasan yang sama tidak dapat memiliki warna yang sama. Masalah pewarnaan peta kemudian adalah mewarnai peta yang diberikan dengan warna sesedikit mungkin. Tidak ada yang pernah menemukan peta yang membutuhkan lebih dari empat warna. Selama lebih dari 100 tahun diperkirakan bahwa tidak ada peta yang membutuhkan lebih dari empat warna, tetapi tidak ada bukti yang benar yang akan datang. Akhirnya, pada tahun 1976, dugaan empat warna ini diselesaikan. Untuk diskusi, lihat Appel dan Haken (1976).

Sekarang diberikan peta geografis kita dapat membuat grafik planar sebagai berikut. Pertimbangkan setiap negara sebagai simpul. Gabungkan dua simpul dengan sisi jika dua negara yang berkorespondensi dengan dua simpul ini memiliki perbatasan yang sama. Jumlah minimum warna yang diperlukan untuk mewarnai peta adalah bilangan kromatik dari graf yang dibuat. Setiap peta memunculkan grafik planar, dan sebaliknya. Dengan demikian teorema empat warna dapat dirumuskan kembali sebagai teorema penting berikut dalam teori graf.

TEOREMA 6.5.5

Bilangan kromatik graf planar tidak boleh lebih dari empat.

6.6 CATATAN DAN REFERENSI

Setiap buku tentang teori graf akan menjadi referensi yang baik untuk graf Eulerian dan Hamiltonian. Buku-buku oleh Behzad et al. (1979), Bondy dan Murty (1976), Chartrand (1977), Deo (1974), Gibbons (1985), Gondran dan Minoux (1984), Harary (1969a), Ore (1963), Roberts (1976, 1978), dan Wilson (1979) adalah beberapa yang standar. Untuk pembahasan algoritma grafik beberapa referensi yang sangat baik adalah buku-buku oleh Aho et al. (1983), Baase (1978), Even (1979), Gondran dan Minoux (1984), Lawler (1976), dan Reingold et al. (1977). Buku-buku karya Minieka (1978), Papadimitriou dan Steiglitz (1982), dan Syslo et al. (1983) berisi diskusi yang rumit dari beberapa algoritma grafik terkenal. Aplikasi teori graf untuk teori pengkodean, riset operasi, ilmu komputer, dan kimia disajikan dalam Deo (1974). Untuk rincian tambahan tentang register geser umpan balik (disebutkan dalam Bagian 5.2), lihat buku oleh Golomb (1967) dan Ronse (1982).

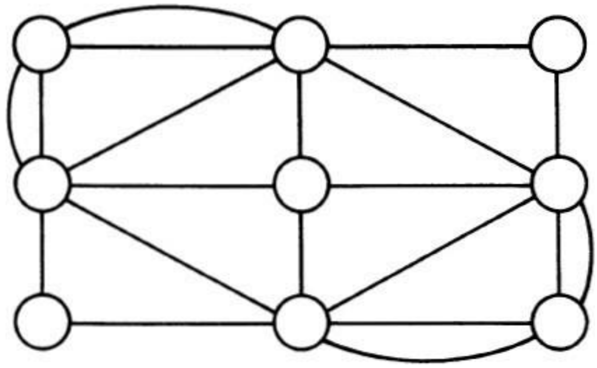
Artikel survei oleh Ralston (1982) tentang barisan de Bruijn sambil menunjukkan hubungan antara teori pengkodean dan teori graf juga menunjukkan bagaimana berbagai bidang matematika diskrit menimpa ilmu komputer. Makalah survei oleh Bellmore dan Nemhauser (1968) tentang Traveling Salesman Problem adalah bacaan pengantar yang baik, dan buku tentang topik yang sama oleh Lawler et al. (1985) adalah studi yang lengkap dan sistematis tentang topik yang terkenal ini. Untuk diskusi yang lebih umum tentang turnamen, lihat buku oleh Moon (1968).

Beberapa referensi yang sangat baik tentang pewarnaan simpul dalam grafik adalah bab-bab yang relevan dalam buku-buku oleh Behzad et al. (1979), Berge (1962), Chartrand (1977), Grimaldi (1985), Gould (1988), Harary (1969a), Liu (1985), Roberts (1976, 1978, 1984), dan Wilson (1979). Cakupan sebanding dari materi yang disajikan dalam bab ini terkandung dalam Bab 3 (Bagian 3 dan 4) dan Bab 6 (Bagian 1) dari Roberts (1984). Lihat makalah oleh Birkhoff dan Lewis (1960) dan Read (1968) untuk bacaan tambahan tentang bilangan kromatik. Dugaan empat warna yang terkenal berasal dari tahun 1852 dan diselesaikan 124 tahun kemudian ketika Appel dan Haken (1976) menunjukkan bahwa setiap peta planar dapat diwarnai dengan empat warna atau lebih sedikit. Lihat makalah Haken (1977) untuk deskripsi bukti teorema ini.

Karena pembuktian mereka bergantung pada pembagian masalah menjadi beberapa kasus tergantung pada susunan negara-negara di peta dan menganalisis berbagai warna susunan ini dengan menulis program komputer, ada kontroversi mengenai sifat pembuktian ini. Lihat makalah oleh Tymoczko (1980) untuk beberapa dasar filosofis mengenai kontroversi ini. Apakah ada bukti matematis murni tanpa menggunakan analisis komputer yang menunjukkan bahwa setiap peta dapat diwarnai dengan empat warna atau lebih sedikit? Ini masih merupakan masalah terbuka. Catatan sejarah yang menarik dari dugaan terkenal ini sebelum pembuktiannya dapat ditemukan pada Mei (1965) dan Harary (1969b).

6.7 LATIHAN

6.1 Temukan sirkuit Euler pada grafik Gambar 6.17



Gambar 6.17 Grafik untuk menentukan sirkuit euler

6.2 Buktikan bahwa suatu graf adalah Euler jika dan hanya jika himpunan sisinya dapat dipartisi menjadi siklus.

6.3 Tunjukkan bahwa digraf terhubung lemah dengan rangkaian Eulerian terhubung kuat.

6.4 Tunjukkan bahwa digraf terhubung lemah dengan lintasan Euler terhubung sepihak.

6.5 Bisakah ada jembatan dalam graf Euler?

6.6 Mungkinkah ada jembatan dalam graf yang memiliki lintasan Euler?

6.7 Buatlah sebuah kata dengan empat huruf A, B, C, dan D menggunakan matriks berikut:

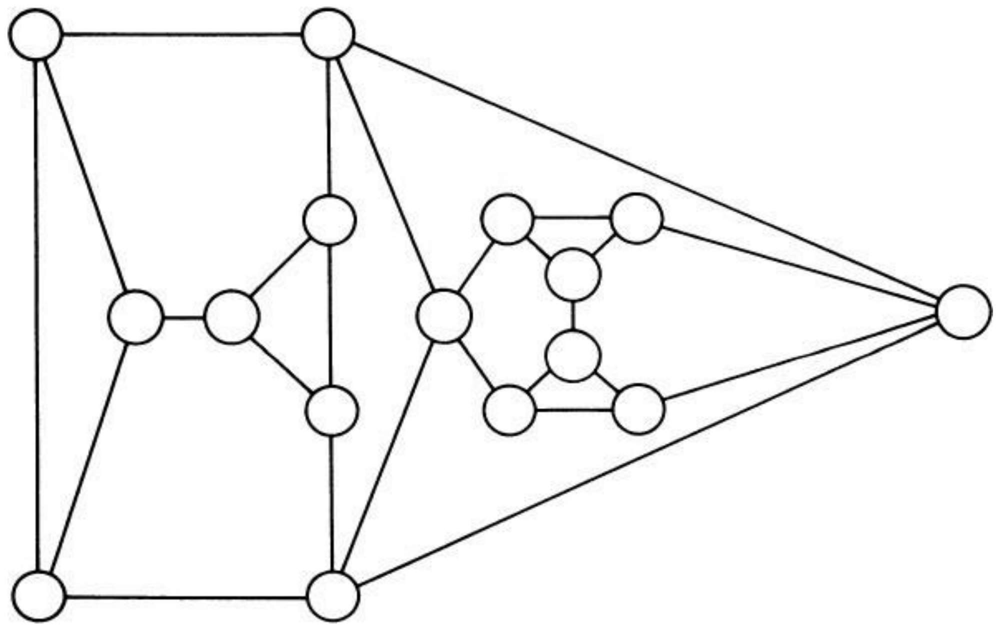
$$
\begin{array}{c c}
 & \begin{array}{cccc} A & B & C & D \end{array} \\
\begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}
\end{array}
$$

6.8 Buatlah sebuah kata dengan empat huruf A, B, C, dan D dengan himpunan frekuensi {2, 1, 2, 4} menggunakan matriks berikut:

$$
\begin{array}{c c}
 & \begin{array}{cccc} A & B & C & D \end{array} \\
\begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}
\end{array}
$$

6.9 Gambarkan digraf de Bruijn G(p, n) dan dapatkan barisan de Bruijn B(p, n) ketika (a) p = 3, n = 2, dan (b) p = 3, n = 3.

6.10 Gambarlah graf terhubung dengan lima simpul yang Eulerian tetapi bukan Hamiltonian.
6.11 Gambarlah graf terhubung dengan empat simpul yang merupakan Hamiltonian tetapi bukan Euler.
6.12 Gambarlah graf terhubung dengan empat simpul yang keduanya Eulerian dan Hamiltonian.
6.13 Gambarlah graf terhubung dengan empat simpul yang bukan Eulerian atau Hamiltonian.
6.14 Buktikan Teorema 6.3.2.
6.15 Buktikan bahwa graf bipartit dengan jumlah simpul ganjil adalah non-Hamilton.
6.16 Jika terdapat lintasan Hamiltonian dari simpul i ke simpul j pada digraf G, maka i adalah “pemenang” dan j adalah “pecundang” di G. Buatlah turnamen dengan lima pemain sedemikian rupa sehingga (a) setiap pemain dapat menjadi pemenang sekaligus pecundang, (b) ada pemenang yang unik dan pecundang yang unik.
6.17 Apakah setiap turnamen terhubung secara sepihak? Apa yang bisa Anda katakan tentang kebalikannya? Membenarkan jawaban Anda.
6.18 Gunakan algoritma terbesar pertama untuk mewarnai simpul dalam grafik Gambar 6.18.



Gambar 6.18 Grafik simpul

BAB 7
POHON DAN APLIKASINYA

7.1 DEFINISI DAN SIFAT-SIFAT

Graf terhubung tanpa siklus adalah pohon. Salah satu struktur diskrit yang banyak dipelajari adalah struktur pohon. Pohon sangat cocok untuk mewakili struktur hierarkis dan alamat serta label. Jenis pohon khusus digunakan dalam teori pengkodean dan dalam pencarian. Kita akan melihat beberapa aplikasi ini dalam bab ini. Sebelum melakukan ini, kami akan menetapkan beberapa hasil yang berkaitan dengan karakterisasi pohon.

1. Dalam sebuah pohon terdapat jalur sederhana yang unik antara setiap pasangan simpul. Kami membuktikan pernyataan ini sebagai berikut. Misalkan u dan v adalah dua simpul pada pohon T. Karena T terhubung, maka ada jalur antara u dan v dan oleh karena itu ada jalur sederhana di antara keduanya. Jika memungkinkan, misalkan P dan P′ adalah dua jalur sederhana di antara keduanya. Jika kedua jalur tidak sama, ada sisi di P yang tidak ada di P′. Mari kita asumsikan bahwa e adalah sisi pertama yang kita temui ketika kita pergi dari u ke v yang ada di P tetapi tidak di P′.

   Membiarkan:

   $$P: \quad u. \; . \; . \; . \; . \; . \; . \; . \; . u_i \overset{e}{- - - -} u_{i+1} . \; . \; . \; . \; . \; . \; . \; . \; . v$$

   dan

   $$P': \quad u. \; . \; . \; . \; . \; . \; . \; . \; . u_i - - - - v_{i+1} . \; . \; . \; . \; . \; . \; . \; . \; . v$$

   Misalkan W adalah himpunan simpul antara di P antara ui+ 1 dan v, dan misalkan W′ adalah himpunan simpul antara di P′ antara vi + 1 dan v. Jika W dan W′ tidak memiliki elemen yang sama, maka kita dapatkan siklus mulai dari ui melalui semua simpul di W, simpul v, dan kemudian semua simpul di W′. Sebaliknya, jika W dan W′ memiliki simpul yang sama, misalkan r adalah subscript terkecil dari simpul ur di P sehingga ur berada di W′. Jadi tidak ada simpul perantara di P antara ui dan ur yang ada di P′. Kemudian kami memiliki siklus mulai dari ui yang melewati semua simpul di W hingga ur dan kemudian semua simpul di W′ dari ur ke ui. Jadi keberadaan dua jalur sederhana yang berbeda antara dua simpul menyiratkan adanya siklus. Menurut definisi, pohon adalah asiklik. Jadi ada jalur unik antara setiap pasangan simpul di pohon.
2. Sebaliknya, jika terdapat lintasan sederhana yang unik antara setiap pasangan simpul pada graf G, maka G adalah pohon. Misalkan G bukan pohon. Maka setidaknya ada satu siklus C di G yang menyiratkan bahwa antara dua simpul di C ada dua jalur sederhana dan ini adalah kontradiksi.

3. Pada pohon T, sisi antara dua simpul v dan w adalah jalur unik di antara keduanya, dan jika kita menghapus sisi ini dari T, maka T tidak terhubung lagi. Dengan kata lain, setiap tepi di pohon adalah jembatan.
4. Sebaliknya, jika G adalah graf terhubung sehingga setiap sisi adalah jembatan, maka G adalah pohon. Misalkan G bukan pohon dan misalkan C adalah sebuah siklus di G. Misalkan e adalah sembarang sisi di C. Misalkan G′ adalah subgraf dari G setelah e. Karena e adalah jembatan, G′ tidak lagi terhubung. Misalkan p dan q adalah sembarang dua simpul di G. Terdapat lintasan P antara p dan q di G. Jika P tidak mengandung e, maka P juga merupakan lintasan antara p dan q pada graf (terputus) G′. Di sisi lain, jika e = (v, w) adalah sisi di P yang juga berada dalam siklus C yang dimulai dari titik t, kita memiliki jalur berikut di G′ antara p dan q:

   p. . . . .v. . . . .t. . . . .w q

   Dengan kata lain, ada jalur antara setiap pasangan simpul di G′, dan ini bertentangan dengan fakta bahwa G′ tidak terhubung.
5. Sebuah pohon T dengan n simpul memiliki (n – 1) rusuk. Kami membuktikan ini dengan induksi pada n. Hal ini benar ketika n = 1. Misalkan itu benar untuk semua m, di mana 1 < m < n. Misalkan e = {u, w} adalah rusuk di T. Karena T adalah pohon, e adalah jembatan. Hapus e untuk mendapatkan subgraf T′ yang memiliki dua komponen terhubung, H dan H′. Baik H dan H′ adalah pohon dengan k dan k′ simpul. Sekarang k dan k adalah bilangan bulat positif yang jumlahnya n. Jadi keduanya kurang dari n. Dengan hipotesis induksi kami H memiliki (k – 1) tepi dan H′ memiliki (k′ – 1) tepi, dan bersama-sama mereka memiliki k + k′ – 2 = n – 2 tepi. Jadi T memiliki (n – 2) tepi, dan akibatnya, T memiliki (n – 1) tepi.
6. Kebalikan dari (5) benar: sembarang graf terhubung G dengan n simpul dan (n – 1) rusuk adalah pohon. Karena jika G = (V, E) bukan pohon, ada sisi e yang bukan jembatan. Kami menghapus e untuk mendapatkan subgraf terhubung G′ = (V, E′). Lanjutkan demikian sampai kita mendapatkan subgraf H = (V, F) di mana setiap sisi adalah jembatan. Jadi H adalah pohon dengan (n – 1) rusuk. Misalkan k (> 0) adalah jumlah sisi yang dihilangkan dari G dalam proses ini. Kita melihat bahwa setelah menghapus k tepi dari (n – 1) tepi kita memiliki (n - 1) tepi!
7. Pernyataan kita selanjutnya adalah bahwa sembarang graf asiklik G = (V, E) dengan n simpul dan (n – 1) rusuk terhubung dan oleh karena itu sebuah pohon. Misalkan G tidak terhubung. Biarkan komponen G menjadi Gi (dengan ni simpul) untuk i = 1, 2, . . . , r. Sekarang setiap komponen Gi adalah asiklik dan terhubung dan oleh karena itu pohon dengan ni – 1 tepi. Jadi jumlah seluruh rusuk pada G = n1 + n2 + • • • + nr – r sama dengan n – r. Tetapi G memiliki tepat n – 1 sisi. Jadi r = 1. Artinya, G memiliki tepat satu komponen asiklik.
8. Misalkan G adalah sembarang graf dengan n simpul. Jika salah satu dari dua pernyataan berikut benar, maka pernyataan ketiga juga benar: (a) G terhubung, (b) G asiklik, dan (c) G memiliki (n – 1) rusuk.
9. Misalkan T adalah sembarang pohon. Gabungkan dua simpul yang tidak berdekatan v dan w dengan sebuah rusuk baru, menghasilkan graf G. Maka G memiliki tepat satu siklus, yang

terdiri dari rusuk baru dan jalur sederhana yang unik di T antara v dan w. Di sisi lain, jika G adalah graf asiklik sedemikian rupa sehingga setiap kali dua simpul yang tidak berdekatan dihubungkan oleh sisi baru, graf yang dihasilkan memiliki tepat satu siklus, maka G adalah pohon. Buktinya adalah dengan kontradiksi. Misalkan G bukan pohon. Kemudian tidak terhubung. Jadi ada sepasang simpul p dan q di G sedemikian rupa sehingga tidak ada jalur di antara mereka sehingga penambahan sisi baru {p, q} tidak membuat siklus.

10. Misalkan G adalah sembarang graf terhubung sehingga setiap kali dua simpul yang tidak berdekatan dihubungkan oleh sisi baru, graf yang dihasilkan memiliki tepat satu siklus. Maka G adalah asiklik dan karenanya merupakan pohon.

Kami sekarang meringkas urutan pernyataan tanpa henti ini sebagai tiga teorema.

TEOREMA 7.1.1

Berikut ini adalah ekuivalen dalam graf sederhana G:

(a) G terhubung dan asiklik.
(b) G terhubung dan jumlah sisi di G kurang satu dari jumlah simpul di dalamnya.
(c) G adalah asiklik dan jumlah sisi di G kurang satu dari jumlah simpul di dalamnya.
(d) G terhubung dan setiap sisi adalah jembatan.
(e) Ada jalur sederhana yang unik antara setiap pasangan simpul di G.
(f) G adalah asiklik dan jika dua simpul yang tidak berdekatan digabungkan untuk membentuk G′, maka G′ memiliki tepat satu siklus.
(g) G terhubung dan jika ada dua simpul yang tidak berdekatan digabungkan untuk membangun G′, maka G′ memiliki tepat satu siklus.

DEFINISI 7.1.1

Subgraf T dari graf G dengan n simpul adalah pohon merentang dalam graf jika (a) T adalah pohon dan (b) T memiliki n simpul.

TEOREMA 7.1.2

Graf G terhubung jika dan hanya jika memiliki pohon merentang.

TEOREMA 7.1.3

Misalkan G adalah graf sederhana dengan n simpul. Jika subgraf H dengan n simpul memenuhi salah satu dari tiga sifat berikut, maka subgraf H juga memenuhi ketiga.

(a) H terhubung.
(b) H memiliki (n – 1) rusuk.
(c) H adalah asiklik.

(Perhatikan bahwa Teorema 7.1.3 mencirikan pohon merentang dalam suatu graf. Dalam Bab 5 kita menggunakan prosedur pencarian kedalaman-pertama untuk mendapatkan pohon merentang dalam graf sederhana yang terhubung.)

7.2 POHON SPANNING

Dalam Bab 4 kami menyebutkan bahwa teori graf "lahir" pada tahun 1736. Dalam nada yang sama dapat dikatakan bahwa pohon pertama kali digunakan oleh G. B. Kirchhoff (1824-1877) pada tahun 1847 dalam karyanya tentang jaringan listrik. Analisis jaringan listrik sebenarnya mengurangi untuk menemukan semua pohon merentang dari grafik yang dipertimbangkan. Spanning tree juga membentuk dasar untuk sejumlah besar masalah dalam optimasi jaringan. Beberapa dari masalah ini dibahas dalam Bab 7. Meskipun Kirchhoff menggunakan pohon dalam analisisnya, Arthur Cayley (1821-1895) yang, satu dekade kemudian, menggunakan pohon secara sistematis dalam usahanya untuk menghitung isomer dari hidrokarbon jenuh (senyawa dalam bentuk Ck H2k +2), yang dapat direpresentasikan sebagai grafik terhubung dengan 3k + 2 simpul (satu untuk setiap atom karbon C dan satu untuk setiap atom hidrogen H). Karena valensi C dan H adalah 4 dan 1, jumlah semua derajat adalah 4k + (2k + 2) dan oleh karena itu harus ada 3k + 1 tepi. Dengan demikian graf yang ditinjau sebenarnya adalah sebuah pohon. Dengan kata lain, graf T suatu hidrokarbon dengan k atom karbon adalah pohon merentang pada graf sembarang dengan 3k + 2 simpul sedemikian rupa sehingga derajat tiap simpul C adalah 4 dan derajat tiap simpul H adalah 1. Pertanyaan yang wajar untuk ditanyakan: Berapa banyak hidrokarbon berbeda yang dapat ada untuk nilai k tertentu? Dalam konteks ini Cayley membuktikan teorema pada jumlah pohon merentang dalam grafik. Sebuah pohon dengan n simpul disebut pohon berlabel jika setiap simpul diberi label unik i di mana i adalah bilangan bulat positif antara 1 dan n. Dua pohon berlabel berbeda jika himpunan tepinya berbeda. Misalnya, 1- - -2- - -3, 1- - -3- - -2, dan 2- - - 1- - -3 adalah tiga pohon berlabel berbeda ketika n = 3, sedangkan 1-- -2- - -3 dan 3- - -2- - -1 bukan pohon berlabel berbeda.

TEOREMA 7.2.1

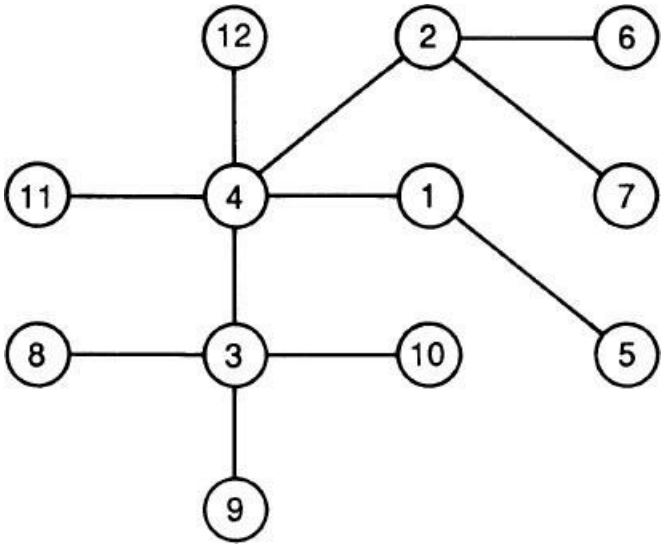
Jumlah pohon berlabel berbeda (dengan n simpul) adalah nn–2 (di mana n paling sedikit 2).

Bukti:

Biarkan N′ himpunan semua (n – 2)-tupel dari N = {1, 2, . . . , n}. Setiap elemen dalam N′ memiliki n – 2 komponen dan setiap komponen dapat dipilih dengan n cara. Jadi kardinalitas dari N′ adalah nn–2 . Teorema kita terbukti jika kita membuat korespondensi satu-satu antara N′ dan himpunan pohon berlabel berbeda dengan n simpul. Misalkan T adalah sembarang pohon berlabel dengan n simpul, dan misalkan W adalah himpunan simpul di T berderajat 1. (Simpul berderajat 1 pada pohon adalah daun.) W memiliki paling sedikit dua dan paling banyak n – 1 elemen. Susun elemen W dalam urutan yang meningkat dan biarkan w1 menjadi elemen pertama di W. Biarkan s1 menjadi simpul unik yang berdekatan dengan w1. Selanjutnya, misalkan T′ adalah pohon yang diperoleh dengan menghapus w1 dari T, dan misalkan W′ adalah himpunan simpul berderajat 1 di T′ yang disusun dalam urutan naik. Jika w2 adalah elemen pertama di W′, kita ambil s2 sebagai simpul unik yang berdekatan dengan w2 di T′. Kami melanjutkan proses ini sampai kami mendapatkan (n – 2)-tupel s dari bentuk (s1 s2 s3 . . . sn–2) menetapkan bahwa setiap pohon berlabel sesuai dengan elemen unik di N′.

Sebelum kita menetapkan hasil dalam arah yang berlawanan, mari kita dapatkan 10-tupel untuk pohon berlabel T dengan 12 simpul seperti yang ditunjukkan pada Gambar 7.1. W = {5, 6, 7, 8, 9, 10, 11, 12} dan s1 = 1, yang merupakan simpul yang berdekatan dengan elemen pertama di W. Hapus dari T simpul 5 dan tepi yang menghubungkan 1 dan 5 ke dapatkan pohon T′ di mana W′ adalah himpunan semua simpul berderajat 1 yang disusun dalam urutan naik. Elemen pertama di W′ adalah 1, jadi s2 adalah 4.

Selanjutnya, hapus simpul 1 dan tepi yang menghubungkan 1 dan 4. Titik pertama berderajat 1 di pohon baru adalah 6, yang berbatasan dengan 2. Jadi s3 adalah 2. Kami melanjutkan dengan cara yang sama dan mengamati bahwa s4 = 2, s5 = 4 , s6 = 3, s7 = 3, s8 = 3, s9 = 4, dan akhirnya, s10 = 4. Jadi pohon berlabel T sesuai dengan 10-tupel (1 4 2 2 4 3 3 3 4 4).



Gambar 7.1 Simpul berdekatan dengan elemen pertama

Selanjutnya, kami membuktikan bahwa setiap (n – 2)-tupel s mendefinisikan pohon berlabel unik dengan n simpul. Jika:

$$s = (s1\ s2\ s3\ .\ .\ .\ sn–2)$$

kita definisikan:
v1 = elemen pertama di N yang tidak ada di s
v2 = elemen pertama di N – {v1} yang tidak ada di s – {s1}
v3 = elemen pertama di N – {v1, v2} yang tidak ada di s – {s1, s2} , dst.
Kami mengulangi proses ini sampai kami mendapatkan v, (i = 1, 2, 3, n – 2). Dua elemen yang tersisa di N dilambangkan dengan x dan y. Sekarang buatlah sebuah graf yang himpunan simpulnya adalah N dengan sisi-sisinya adalah (n – 2) sisi yang menghubungkan si dan vi dan sisi yang menghubungkan x dan y. Grafik yang diperoleh adalah pohon berlabel unik yang sesuai dengan s.
Misalnya, jika:

$$s = (1\ 4\ 2\ 2\ 4\ 3\ 3\ 3\ 4\ 4)$$

maka n = 12 dan N = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}. Jadi v1 = elemen pertama di N yang tidak ada di s. Jadi v1 = 5. Selanjutnya v2 = elemen pertama di N – {5} yang tidak ada di s – {1}. Jadi v2 = 1. v3 = elemen pertama di N – {5, 1} yang tidak ada di s - {1, 4}. Jadi v3 = 6. Lanjutkan seperti ini, kita mendapatkan v4 = 7, v5 = 2, v6 = 8, v7 = 9, v8 = 10, v9 = 3, dan v10 = 11. Akhirnya, x = 4 dan y = 12. Sekarang buatlah graf dengan himpunan simpul N dan sisi yang menghubungkan si dan vi (i = 1, 2, 3, , 10) dan sisi yang menghubungkan x dan y.

Grafik ini persis pohon berlabel pada Gambar 7.1. Jadi korespondensi satu-satu antara N′ dan himpunan pohon berlabel berbeda dengan n simpul ditetapkan untuk membuktikan teorema tersebut.

Catatan: Teorema Cayley adalah teorema keberadaan. Teorema ini tidak menyelesaikan masalah menemukan semua pohon berlabel berbeda.

7.3 POHON BINER

Digraf disebut pohon berarah jika graf yang mendasarinya adalah pohon. Pohon berarah adalah pohon berakar jika (1) terdapat tepat satu simpul (disebut akar) dengan derajat masuk 0 dan (2) derajat masuk setiap simpul lainnya adalah 1. Suatu simpul pada pohon berakar adalah simpul terminal (atau daun) jika derajat keluarnya adalah 0. Sebuah simpul nonterminal disebut simpul menengah atau dalam. Akar dengan demikian merupakan simpul perantara. Banyaknya busur pada lintasan dari akar ke suatu simpul disebut tingkat simpul tersebut. Menurut definisi level akarnya adalah 0. Jika levelnya adalah 0, 1, 2, . . . , k, maka k adalah tinggi pohon. Simpul v adalah turunan dari simpul u jika ada busur dari u ke v. Jika T adalah pohon berakar, subpohon T' pada simpul v adalah pohon berakar T' = (V', E') sedemikian sehingga (1) akar dari T' ada di v, (2) V' terdiri dari v dan semua turunannya di T, dan (3) E' berisi semua busur dari semua jalur berarah (di T) dari v ke semua daun . Sebuah pohon dipangkas di simpul v jika kita menghapus semua turunan dari v dan semua busur dari semua jalur berarah yang berasal dari v sehingga v menjadi daun dari pohon yang dipangkas itu.

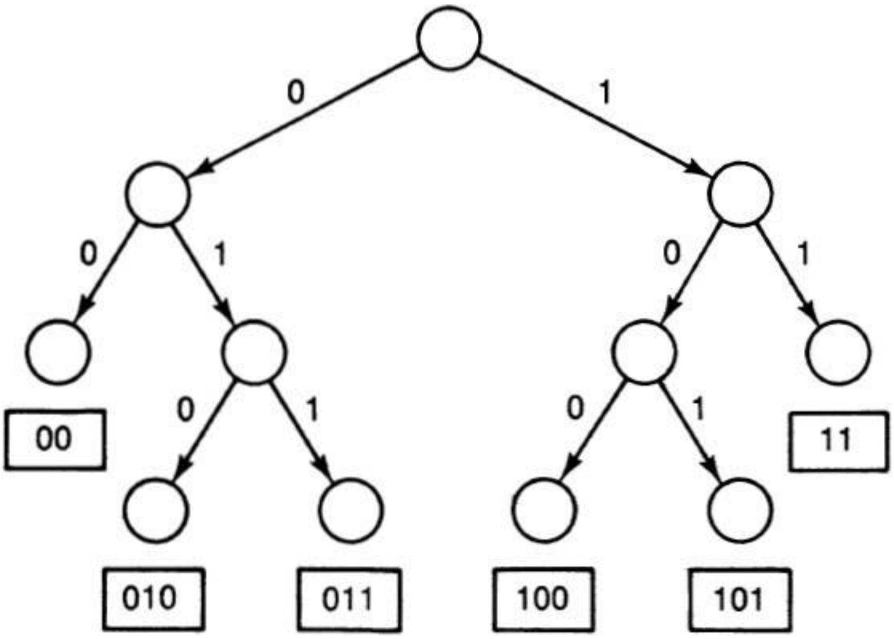
Sebuah pohon berakar adalah pohon biner jika derajat keluar setiap simpul menengah paling banyak 2. Pada pohon biner beraturan derajat keluar setiap simpul menengah tepat 2. Sebuah pohon biner beraturan penuh jika semua daunnya berada pada tingkat yang sama. Banyak struktur diskrit dapat direpresentasikan sebagai pohon biner. Kami membahas di sini aplikasi dalam teori pengkodean.

Dalam komputer, karena semua informasi disimpan dalam bentuk bilangan biner, setiap kali huruf dalam alfabet atau simbol dimasukkan ke dalam komputer, itu diubah menjadi kata biner melalui kode karakter yang merupakan kode satu-ke- satu korespondensi antara himpunan semua karakter dan himpunan bilangan biner. Misalnya, dua kode karakter yang paling umum digunakan adalah ASCII (American Standard Code for Information Interchange) dan EBCDIC (Extended Binary-Coded Decimal Interchange Code). Dalam kedua kode ini, setiap karakter diberi kode dengan panjang tetap dan oleh karena itu mereka dikenal sebagai kode karakter dengan panjang tetap. Dalam ASCII panjangnya biasanya 8, dan 6 di EBCDIC. Jadi untuk mengkodekan angka 247 kita menggunakan pesan 24 bit: 011100100111010001110111. Decoding juga mudah. Jumlah bit dalam pesan adalah kelipatan dari 8 dan kami menggunakan kode secara sistematis untuk menguraikan pesan.

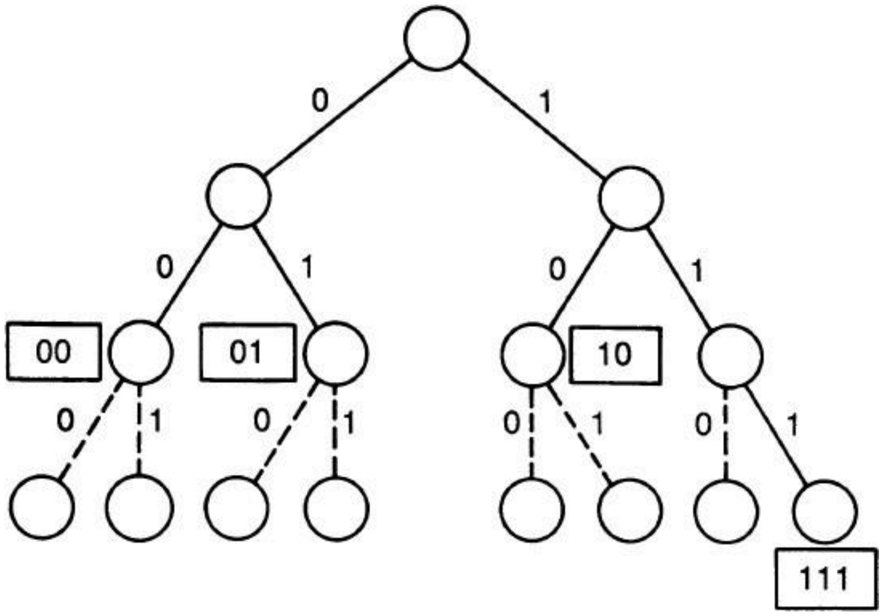
Kelemahan utama dari kode karakter dengan panjang tetap adalah bahwa semua karakter, apakah mereka sering digunakan atau tidak, membutuhkan jumlah bit yang sama. Hal ini tentu menguntungkan untuk memiliki kode di mana karakter yang lebih sering digunakan menggunakan bit yang lebih sedikit sehingga panjang total pesan sekecil mungkin. Dengan kata lain, kode karakter dengan panjang variabel lebih tepat. Tetapi saat menggunakan kode seperti itu, kita harus memastikan bahwa decoding tidak ambigu. Misalnya, jika kode untuk 1, 2, dan 3 berturut-turut adalah 01, 0, dan 00, maka kata 0001 dapat didekode menjadi 221 atau 31. Alasan yang jelas untuk ambiguitas ini adalah kenyataan bahwa kode untuk 2 muncul sebagai bagian dari kode untuk nomor lain di awal. Sebuah kata w adalah awalan dari kata lain v jika v = wp, di mana p adalah kata lain. Sebuah kode karakter dikatakan memiliki properti awalan jika tidak ada kode untuk suatu simbol yang merupakan awalan dari kode untuk simbol yang lain. Kode karakter adalah kode awalan jika memiliki properti awalan.

Sangat mudah untuk melihat bahwa kode awalan biner dapat diperoleh langsung dari pohon biner biasa. Pertama kita memberi label pada dua busur yang datang dari sebuah simpul perantara sebagai 0 dan 1. Kemudian berikan ke setiap daun urutan bit yang merupakan urutan label busur dari akar ke daun itu. Kode karakter yang menetapkan setiap karakter ke dalam urutan yang sesuai dengan daun tentu merupakan kode awalan. Misalnya, penugasan A = 00, B = 010, C = 011, D = 100, E = 101, dan F = 11 yang diperoleh dari pohon biner beraturan Gambar 6.3.1 merupakan kode awalan untuk alfabet {A, B, C, D, E, F}.

Sebaliknya, sesuai dengan kode awalan yang diberikan, kita dapat membangun pohon biner reguler dengan memangkas pohon biner reguler penuh dengan panjang k, di mana k adalah panjang urutan terbesar dalam kode. Misalnya, perhatikan kode R = 00, E = 01, A = 10, dan D = 111 dengan properti awalan. Di sini k = 3 dan pertama-tama kita membangun pohon biner reguler penuh dengan tinggi 3 dan memberi label setiap busur yang berasal dari setiap simpul perantara sebagai 0 atau 1. Kemudian menetapkan ke setiap simpul pohon urutan bit yang merupakan urutan label dari busur dari akar ke simpul itu. Jadi setiap barisan biner dengan panjang kurang dari atau sama dengan k ditetapkan ke simpul unik. Temukan simpul-simpul yang urutan binernya sama persis dengan urutan dalam kode awalan yang diberikan dan pangkas pohon di simpul-simpul ini. Hapus juga daun yang tidak sesuai dengan urutan dalam kode. Pohon biner pada Gambar 7.2 diperoleh untuk kode awalan yang diberikan setelah menemukan simpul yang sesuai.



Gambar 7.2 Pohon biner



Gambar 7.3 Pohon biner

Dengan demikian diberi kode awalan, string arbitrer dapat dengan jelas didekodekan dengan melanjutkan dari kiri ke kanan dalam string, menemukan substring pertama yang merupakan karakter, kemudian substring berikutnya, dan seterusnya. Misalnya, jika kode awalan adalah 00, 010, 011, 100, 101, dan 11, penguraian unik dari string 1011110010000010011110 adalah 101, 11, 100, 100, 00, 010, 011, 11. Dua bit terakhir adalah tidak digunakan dalam kasus ini.

Akhirnya, untuk alfabet tertentu dimungkinkan untuk memiliki lebih dari satu kode awalan. Misalnya, {00, 01, 10, 11} dan {0, 11, 100, 101} adalah kode awalan untuk {A, B, C, D}. Jadi wajar untuk bertanya apakah satu kode "lebih baik" dari yang lain. Dalam konteks inilah kami mempertimbangkan efisiensi kode.

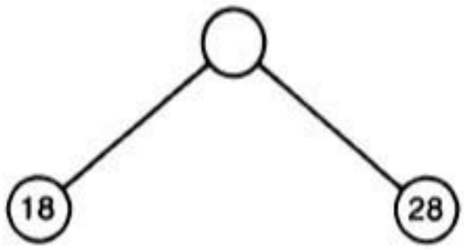
Misalkan {a1, a2, . . . , an} menjadi alfabet dan biarkan fi menjadi frekuensi kemunculan ai secara rata-rata. Biarkan C menjadi kode awalan apa pun untuk alfabet dan biarkan li menjadi panjang kode untuk ai. Bobot w(C) dari kode C adalah f1l1 + . . . + fln. Masalahnya adalah menemukan kode C sedemikian rupa sehingga w(C) sekecil mungkin dan setara dengan masalah berikut: Mengingat bahwa fi (i = 1, 2, . . , n) adalah frekuensi dari karakter ai dalam alfabet n huruf, temukan pohon biner beraturan dengan n simpul terminal sedemikian rupa sehingga f1l1 + f2l2 + . . . + fnln adalah minimum di mana li adalah panjang jalan dari akar ke simpul terminal ke-i, yang mewakili ai. Pohon seperti itu dikenal sebagai pohon biner optimal untuk distribusi frekuensi yang diberikan. Kami sekarang membahas prosedur elegan karena D. Huffman (1952) untuk mendapatkan pohon seperti itu.

Pertama kita atur frekuensi dalam urutan yang tidak menurun dari f1 ke fn. Algoritma Huffman didasarkan pada fakta berikut (lihat Teorema 7.3.1 di bawah): Jika T’ adalah pohon optimal yang diperoleh dengan menggunakan prosedur ini untuk frekuensi (k – 1) {f1 + f2, f3, . . . , fk} dengan (k – 1) simpul terminal, pohon T diperoleh dari T dengan memperkenalkan dua simpul terminal baru v1 (untuk mewakili f1) dan v2 (untuk mewakili f2) dan dengan menggabungkan simpul terminal v di T” yang mewakili f1 + f2 hingga v1 dan v2 adalah pohon optimal untuk himpunan {f1, f2, f3, . . . , fk}. (Jumlah f1 dan f2 tidak perlu kurang dari f3.

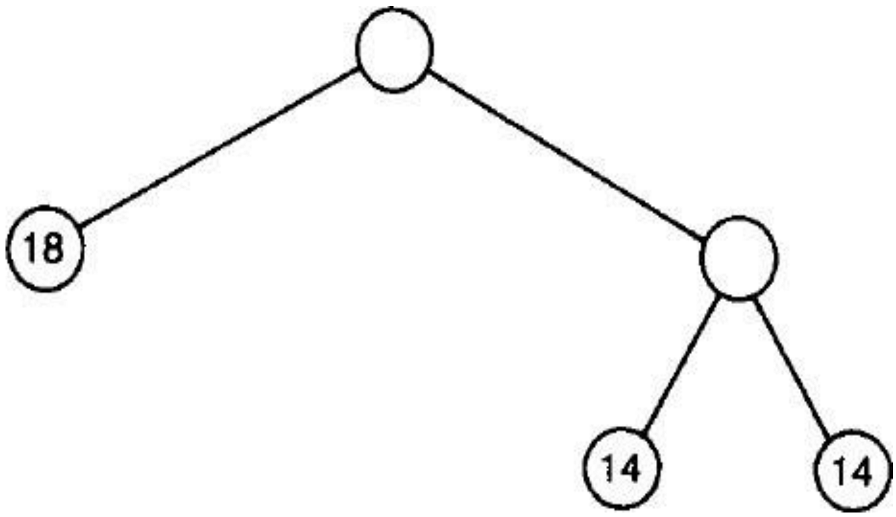
Tiebreaking bersifat arbitrer dalam arti bahwa jika f1 + f2 = f3, maka v dapat berupa simpul yang menyatakan f1 + f2 atau simpul yang menyatakan f3. Perhatikan bahwa v bukan simpul terminal di T.)

Berikut adalah contoh yang menggambarkan konstruksi kode Huffman untuk alfabet enam huruf dengan distribusi frekuensi S1 = {3, 5, 6, 8, 10, 14}. Tambahkan dua angka pertama dan atur ulang untuk mendapatkan S2 = {6, 8, 8, 10, 14}. Ulangi proses ini untuk mendapatkan S3 = {8, 10, 14, 14}, S4 = {14, 14, 18}, dan S5 = {18, 28}. Sekarang buat pohon optimal untuk S5 seperti pada Gambar 7.4. Sekarang frekuensi 28 adalah jumlah dari 14 dan 14 di S4, dan ini akan membawa kita ke S4 dan pohon pada Gambar 7.5. Kemudian kita menggunakan dekomposisi 18 = 8 + 10 untuk mendapatkan pohon untuk S3 seperti pada Gambar 7.6. Setelah itu kita menggunakan dekomposisi 14 = 6 + 8 (sekali) dan 8 = 3 + 5 untuk mendapatkan pohon Huffman yang diinginkan, seperti pada Gambar 7.6.

Jadi kode Huffman untuk distribusi frekuensi {3, 5, 6, 8, 10, 14} adalah {000, 001, 110, 111, 01, 10} dengan panjang 3, 3, 3, 3, 2, 2, dan bobot kode ini adalah 3.3 + 3.5 + 3.6 + 3.8 + 2.10 + 2.14 = 114, yang merupakan minimum.



Gambar 7.4 Pohon biner



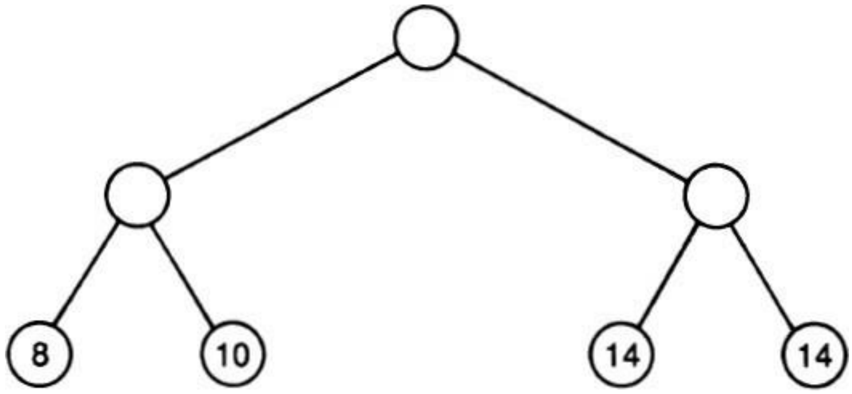
Gambar 7.5 Pohon biner

TEOREMA 7.3.1

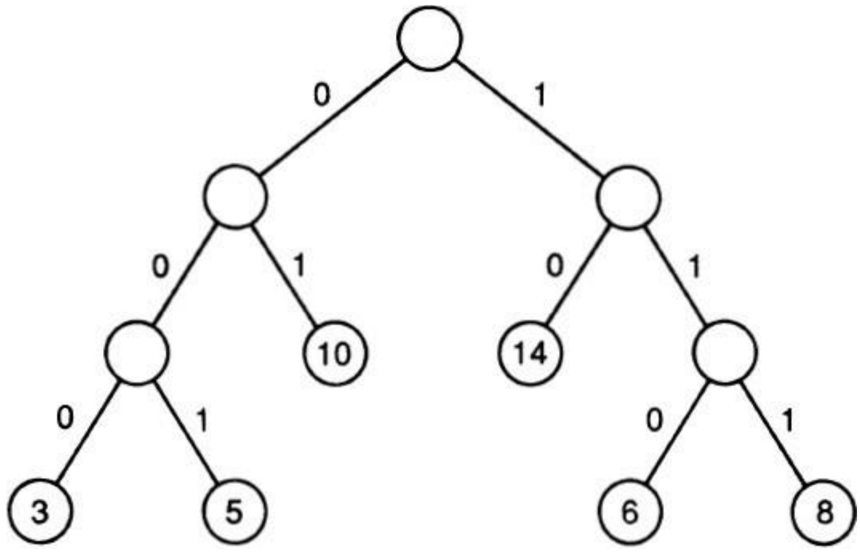
Pohon biner reguler yang diperoleh dengan menggunakan algoritma Huffman adalah pohon optimal.

Bukti:

Biarkan frekuensi n fi (i = 1, 2, . . , n) berada dalam urutan yang tidak menurun. Jumlah pohon biner beraturan dengan n simpul terminal yang mewakili n frekuensi ini berhingga, dan oleh karena itu ada pohon biner beraturan T′ dengan n simpul yang bobotnya w(T′) adalah minimum. Biarkan v menjadi simpul internal dari T′ sedemikian rupa sehingga jarak (yaitu, jumlah tepi di jalan) dari akar pohon ke v tidak kurang dari jarak dari akar ke simpul nonterminal lainnya, dan biarkan keturunannya dari v menjadi simpul terminal yang mewakili fi dan fj . Sekarang tetapkan nilai f1 ke simpul yang mewakili fi dan nilai fi ke simpul yang mewakili fj. Pada saat yang sama, tetapkan nilai fi ke simpul yang mewakili fi dan nilai fj ke simpul yang mewakili f2. Bobot w(T) dari pohon T yang dihasilkan tidak boleh lebih dari w(T″) karena pilihan v. Pada saat yang sama w(T) tidak boleh kurang dari w(T″) karena T″ adalah optimal pohon. Oleh karena itu kami menyimpulkan bahwa ada pohon T optimal yang f1 dan f2 adalah keturunan dari simpul nonterminal yang sama yang berada pada jarak maksimal (dibandingkan dengan simpul nonterminal lainnya) dari akar T.



Gambar 7.6 Pohon biner



Gambar 7.7 Pohon biner

Selanjutnya, misalkan T″ adalah pohon biner beraturan dengan (n – 1) simpul terminal yang diperoleh dari pohon T yang diperoleh di atas dengan menghapus dua simpul terminal yang sesuai dengan f1 dan f2, sehingga simpul v (dari mana kedua simpul ini turun) menjadi simpul terminal, yang sesuai dengan frekuensi f1 + f2 . Sangat mudah untuk melihat bahwa w(T) = w(T″) + f1 + f2 . T optimal jika dan hanya jika T″ optimal. Kesimpulan dari teorema berikut dengan induksi pada n.

Kami menyimpulkan diskusi ini dengan menetapkan bahwa algoritma Huffman memang memberikan sistem pengkodean yang memiliki properti yang diinginkan: Semakin besar frekuensi suatu karakter, semakin pendek panjang kode yang mewakili karakter tersebut.

TEOREMA 7.3.2

Misalkan T adalah pohon Huffman optimal untuk n karakter ai dengan frekuensi fi, dan misalkan di adalah panjang kode yang merepresentasikan ai (i = 1, 2, . . , n). Jika fi < fj, maka di dj.

Bukti:

Panjang dari akar T ke simpul terminal yang merepresentasikan ai adalah dj. Biarkan simpul terminal yang mewakili fi diberi kode ai, dan biarkan simpul terminal yang mewakili fj diberi kode ai. Meskipun pohonnya sama, beratnya akan berubah. Biarkan w menjadi bobot lama dan w adalah bobot baru. Tentu saja, w w'. Sekarang w – w' = (di – dj)(fi – fj) . Jadi di dj.

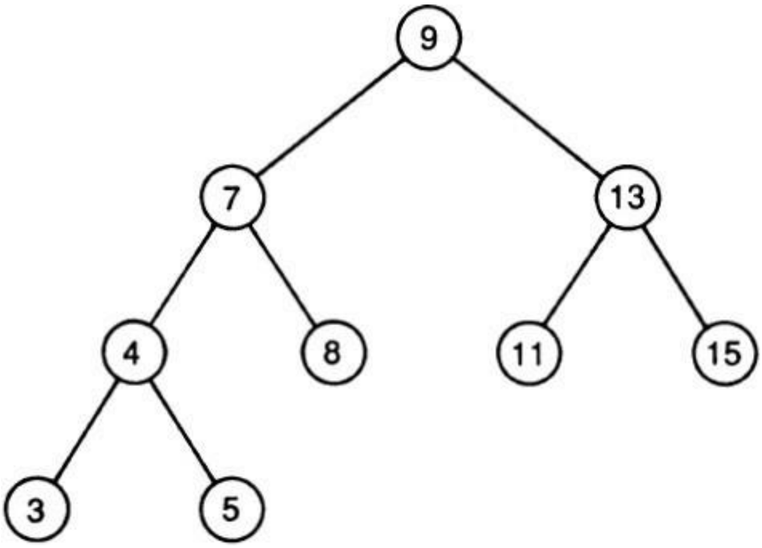
Pohon Pencarian Biner

Operasi penting yang sering muncul dalam ilmu komputer adalah penggunaan pohon biner untuk mencari melalui daftar atau tabel, yang elemen-elemennya membentuk himpunan hingga yang terurut secara linier. Jika T adalah pohon biner dan jika derajat keluar dari simpul v adalah 2, maka v memiliki dua turunan—keturunan kiri dan turunan kanan. (Jika derajat keluar v adalah 1, turunan unik dapat dianggap sebagai turunan kiri atau turunan kanan.) Pohon yang berakar pada turunan kiri suatu simpul disebut subpohon kiri dari simpul ini. Pohon yang berakar pada turunan kanan suatu simpul disebut subpohon kanan dari simpul ini.

Misalkan T adalah pohon biner. Untuk setiap simpul v dari pohon, sebuah bilangan real k(v) diberikan. Bilangan k(v) ini disebut kunci dari simpul. Tetapkan kunci ke simpul T sedemikian rupa sehingga kunci simpul adalah (1) lebih besar dari kunci simpul di subpohon kirinya dan (2) lebih kecil dari kunci simpul di subpohon kanannya. Pohon biner dengan kunci yang ditentukan oleh aturan ini disebut pohon pencarian biner. Pada Gambar 7.8 kita memiliki pohon pencarian biner dengan sembilan simpul di mana himpunan kuncinya adalah {9, 7, 13, 4, 8, 11, 15, 3, 5}.

Jika T adalah pohon biner arbitrer, dimungkinkan untuk menetapkan kunci ke simpul T sedemikian rupa sehingga T adalah pohon pencarian biner. Hal ini dibuktikan dalam Reingold (1977), di mana algoritma untuk mengubah pohon biner menjadi pohon pencarian biner juga diturunkan. Untuk mencari apakah item q tertentu ada dalam daftar, kita menggunakan pohon pencarian biner dari daftar sebagai berikut: Pertama kita mulai dengan root. Pada setiap tahap, kunci dari simpul x diperiksa. Pada awalnya, x adalah akarnya. Jika q = k(x), kami menemukan item yang diinginkan dalam daftar. Jika q < k(x), kita mengabaikan subpohon kanan yang berakar pada x dan melihat turunan kiri dari x. Jika q > k(x), kita abaikan subpohon kiri yang berakar pada x dan lihat turunan kanan dari JC. Kami melanjutkan proses ini sampai kami mencapai pohon dengan satu simpul.

Dalam proses pencarian ini, jumlah perbandingan yang akan dibuat adalah h + 1 dalam kasus terburuk di mana h adalah ketinggian pohon pencarian biner. Dengan demikian kompleksitas komputasi dari algoritma ini diminimalkan jika kita dapat menemukan pohon pencarian biner dengan tinggi minimum. Dalam konteks ini, teorema berikut sangat berguna.



Gambar 7.8 Pohon biner

TEOREMA 7.3.3

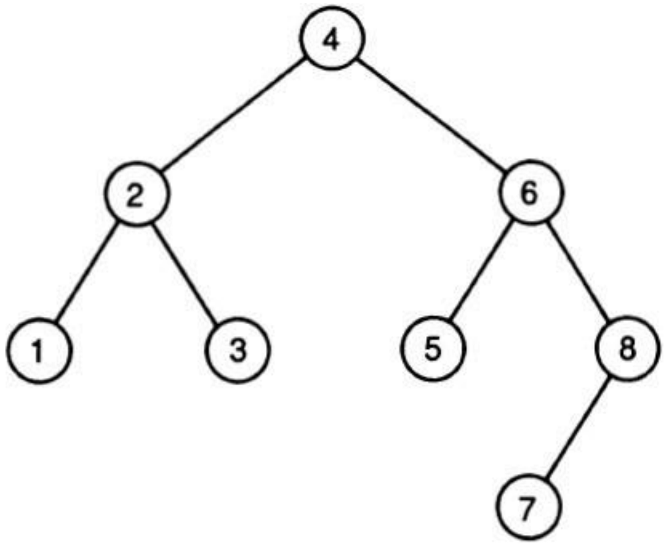
Tinggi minimum pohon biner dengan n simpul adalah m – 1 di mana m adalah langit-langit log(n + 1).

Bukti:

Misalkan T adalah sembarang pohon biner dengan n simpul, dan misalkan h adalah tingginya. Ada paling banyak 2k simpul pada tingkat k, di mana k = 0, 1, 2, . . . , h. Jadi n 1 + 2 + 22 + 23 + . . . + 2j = 2j + 1 – 1

Oleh karena itu h + 1 log(n +1), yang menyiratkan bahwa h m – 1, di mana m adalah langit-langit log(n + 1).

Prosedur rekursif dapat digunakan untuk membangun pohon pencarian biner yang sesuai dengan daftar yang diurutkan secara linier sebagai berikut. Mulailah dengan pohon yang hanya berisi satu simpul, yaitu akar. Kami menetapkan file pertama dalam daftar sebagai kunci dari simpul ini. Untuk menambahkan file baru dari daftar, file ini pertama dibandingkan dengan kunci dari simpul yang ada di pohon, mulai dari akar dan pindah ke kiri jika file kurang dari kunci dari simpul yang dipertimbangkan dan jika ini simpul memiliki turunan kiri, atau bergerak ke kanan jika file lebih besar dari kunci simpul yang dipertimbangkan dan jika simpul ini memiliki turunan kanan. Ketika file saat ini kurang dari kunci sebuah simpul dan jika simpul ini tidak memiliki turunan kiri, simpul baru dengan file ini sebagai kuncinya dimasukkan sebagai turunan kiri baru. Demikian pula, ketika sebuah file lebih besar dari kunci sebuah simpul dan jika simpul ini tidak memiliki turunan kanan, sebuah simpul baru ditambahkan dengan file ini sebagai kuncinya sebagai turunan kanan baru. Prosedur ini diilustrasikan dalam contoh berikut.



Gambar 7.9 Pohon biner

Contoh 7.3.1

Bentuklah derajat pencarian biner untuk mewakili {4, 6, 2, 8, 5, 3, 7, 1}.

Solusi Pohon pencarian biner ditampilkan pada Gambar 6.3.8. Kunci dari root adalah 4, yang merupakan elemen pertama dalam himpunan. Elemen berikutnya, 6, lebih dari 4. Karena 4 tidak memiliki turunan kanan pada tahap ini, sebuah simpul baru dengan kunci 6 dibuat sebagai turunan kanan dari 4. Elemen berikutnya adalah 2. Kita mulai dengan akar dan bergerak ke kiri sejak 2 < 4. Karena 4 tidak memiliki turunan kiri, sebuah simpul baru dibangun sebagai turunan kiri dari 4, dan simpul ini memiliki kunci 2. Elemen berikutnya adalah 8. Bandingkan 8 dengan 4 dan kita bergerak ke kanan. Bandingkan 8 dengan 6 dan kita bergerak ke kanan. Sekarang simpul dengan kunci sebagai 6 tidak memiliki keturunan yang tepat. Sekarang simpul baru dengan kunci 8 dibangun sebagai turunan kanan dari simpul dengan kunci 6. Kami melanjutkan proses ini sampai kami mencapai item terakhir dalam daftar.

7.4 CATATAN DAN REFERENSI

Beberapa referensi umum yang berguna tentang pohon dan propertinya adalah bagian yang sesuai dari buku oleh Behzad et al. (1979), Berge (1962), Bondy dan Murty (1976), Deo (1974), Gondran dan Minoux (1984), Gould (1988), Grimaldi (1985), Harary (1969a), Liu (1985), Roberts (1984), Townsend (1987), Tucker (1984), dan Wilson (1979). Dalil 8.2.1 adalah karena Arthur Cayley (1821-1895), yang menggunakan teori graf sehubungan dengan masalah enumerasi dalam kimia fisik. Untuk perlakuan tambahan kode Huffman, lihat Huffman (1952), Markowsky (1981), dan Standish (1980). Lihat Bab 2 dari Knuth (1973a) dan Bab 6 dari Knuth (1973b) untuk perawatan lengkap pohon dan pohon pencarian.

7.5 LATIHAN

7.1 Jika G adalah hutan dengan n simpul, m sisi, dan k komponen, dapatkan ekspresi untuk m dalam suku n dan k.

7.2 Misalkan sebuah pohon memiliki dua simpul berderajat 5, tiga simpul berderajat 4, enam simpul berderajat 3, delapan simpul berderajat 2, dan r simpul berderajat 1. Temukan r.

7.3 G adalah graf terhubung dengan 20 simpul. Tentukan jumlah minimum rusuk yang dapat dimiliki G.
7.4 G adalah graf terhubung dengan 20 rusuk. Tentukan jumlah maksimum simpul yang dapat dimiliki G.
7.5 Misalkan G memiliki empat komponen, 20 sisi, dan r simpul, Tentukan nilai maksimum r.
7.6 Tunjukkan bahwa setiap pohon adalah graf bipartit. Pohon manakah yang merupakan graf bipartit lengkap?
7.7 Sisi e di G ada di setiap pohon merentang di G. Apa yang dapat Anda katakan tentang e?
7.8 Sisi e di G tidak ada dalam pohon merentang dari G. Apa yang dapat Anda katakan tentang e?
7.9 Jika T dan T' adalah dua pohon merentang di G, apakah T dan T' perlu memiliki sisi yang sama? Buktikan ini atau buat contoh tandingan.
7.10 Tunjukkan bahwa suatu graf terhubung yang setiap simpulnya genap harus memiliki siklus.
7.11 Jika e adalah sisi dalam graf terhubung G (tanpa loop), maka buktikan bahwa terdapat pohon merentang T(e) yang memuat e.
7.12 Jika e dan f adalah dua sisi dalam graf sederhana, ada pohon merentang T(e, f) yang memuat e dan f.
7.13 Temukan pohon berlabel unik yang sesuai dengan s = (8 8 7 7 7 6 6).
7.14 Jika tepi pohon berlabel adalah {1, 2}, {2, 3}, {2, 4}, {4, 5}, {4, 7}, {5, 6}, {7, 8}, dan {8, 9}, cari s.
7.15 Gunakan kode awalan A = 000, B = 001, C = 01, D = 10, E = 111, dan R = 110 untuk memecahkan kode kata berikut: 000001110000010001000000111000011
7.16 Jika frekuensi enam huruf dalam kode awalan pada Soal 6.15 adalah 8, 10, 4, 5, 12, dan 10, tentukan bobot kode tersebut.
7.17 Dapatkan kode awalan optimal untuk data pada Soal 6.16 dan temukan bobot kode ini. Encode kata yang muncul pada Soal 6.15 menggunakan kode ini. Berapa panjang kata ini jika kita menggunakan kode optimal ini dalam kasus terburuk?
7.18 Dekode kata 11111011001001 menggunakan kode optimal yang diperoleh pada Soal 6.17.
7.19 Temukan pohon biner beraturan dengan tinggi k dengan 13 simpul sedemikian rupa sehingga (a) k sekecil mungkin, dan (b) k sebesar mungkin.
7.20 Sebuah pohon berakar dengan tinggi k dikatakan sebagai pohon seimbang jika setiap simpul terminal berada pada level k atau (k – 1). Bangun pohon biner seimbang dengan n simpul ketika n = 11 dan n = 12.
7.21 Buat pohon pencarian biner (menggunakan urutan abjad) untuk himpunan {Hongaria, Jerman, Polandia, Bulgaria, Rumania, Cekoslowakia, Albania, Yugoslavia}.

BAB 8
MASALAH POHON RENTANG

8.1 LEBIH LANJUT TENTANG SPANNING TREES

Jika G adalah graf terhubung dengan n simpul, pohon merentang di G, seperti yang kita lihat pada Bab 6, adalah subgraf asiklik dari G dengan (n - 1) tepi. Jika T = (V, E′) adalah pohon merentang di G = (V, E), sisi-sisi dari G yang tidak berada di E′ disebut tali busur dari T. Jika e adalah tali busur yang menghubungkan simpul u dan v, tepi di jalur unik di T antara u dan v bersama-sama dengan tepi e membentuk siklus unik di G yang disebut siklus fundamental G relatif terhadap T terhadap tali busur e dan dilambangkan dengan CT(e). Jadi, jika G memiliki m tepi, ia akan memiliki m – (n – 1) siklus fundamental tersebut relatif terhadap setiap pohon merentang. Kami berasumsi bahwa G adalah graf terhubung di seluruh bab ini kecuali dinyatakan lain.

Suatu himpunan bagian D dari himpunan sisi-sisi suatu graf G = (V, E) disebut himpunan terputus dari G jika penghapusan sisi-sisi di D dari G membuat G menjadi graf tak-berhubungan. Jika V dipartisi menjadi dua himpunan V′ dan V″ dan jika D = (V′, V″) adalah himpunan semua sisi di E berbentuk {i, j}, di mana i V′ dan j V″, maka D adalah himpunan pemutus.

Himpunan pemutus D disebut himpunan pemutus dari G jika tidak ada himpunan bagian yang tepat dari D yang merupakan himpunan pemutus. Satu set pemutus terdiri dari tepat satu sisi, tentu saja, adalah sebuah potongan yang dikenal sebagai jembatan.

Jika D adalah sebuah cutset di G, penghapusan edge di D memutuskan G menjadi tepat dua komponen G′ (dengan V′ sebagai himpunan simpul) dan G″ (dengan V″ sebagai himpunan simpul), dan dengan demikian D = (V′, V″). Sebaliknya, jika V dipartisi sewenang-wenang menjadi dua himpunan W dan W′, himpunan pemutus D = (W, W′) tidak perlu berupa cutset. Misalnya, misalkan V = {a, b, c}, E = {{a, b}, {a, c}}, W = {a}, dan W = {b, c}. Dalam hal ini himpunan pemutus D = (W, W′) bukan merupakan himpunan potong. Jadi kapan sebuah partisi dari himpunan vertex memunculkan sebuah cutset?

TEOREMA 8.1.1

Jika himpunan simpul V dari graf terhubung G dipartisi menjadi dua himpunan bagian W dan W sedemikian sehingga setiap dua simpul di W (dan W) dihubungkan oleh sebuah lintasan yang hanya terdiri dari simpul-simpul dari W (dan W′), maka D = (W, W′) adalah sebuah cutset.

Bukti:

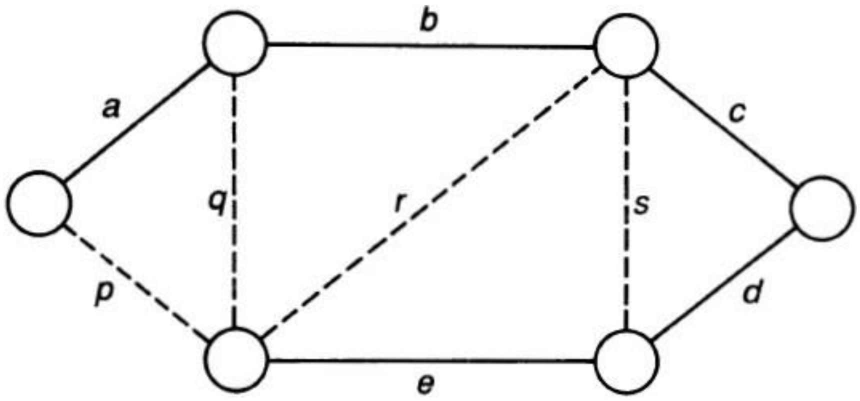
Misalkan D bukan cutset. Maka ada himpunan bagian yang tepat D′ dari D yaitu himpunan bagian. Misalkan e = {w, w′} adalah rusuk di D yang bukan di D′, di mana w di W dan w′ di W′. Misalkan u dan v adalah sembarang dua simpul di G, di mana u di W dan v di W′. Secara hipotesis terdapat lintasan antara u dan w yang terdiri dari simpul-simpul dari W saja, dan ada lintasan antara w′ dan v yang terdiri dari simpul-simpul dari W saja. Jadi ada lintasan antara u dan v menggunakan sisi e yang tidak berada di D′. Jadi D′ bukan himpunan pemutus, yang merupakan kontradiksi.

Akibat Wajar:

Jika T adalah sembarang pohon merentang di G = (V, E), penghapusan setiap sisi di T membuat T terputus dengan membuat dua subpohon dengan himpunan simpul W dan W′ sedemikian rupa sehingga D(W, W′) adalah himpunan bagian dari G .

Jadi sesuai dengan setiap sisi e dari pohon merentang T, ada sebuah potongan unik DT(e) yang disebut potongan dasar dari T sehubungan dengan sisi e. Dengan demikian, setiap graf terhubung dengan n simpul akan memiliki sistem (n – 1) potongan-potongan fundamental terhadap setiap pohon merentang. Misalnya, dalam grafik Gambar 8.1 kita memiliki pohon merentang T dengan tepi a, b, c, d, dan e. Kuncinya adalah p, q, r, dan s. T dengan demikian memiliki empat siklus dasar (satu terhadap setiap akord T) dan lima potongan dasar (satu sehubungan dengan setiap tepi T). Kita lihat bahwa rusuk-rusuk dalam siklus fundamental C(r) adalah r, c, d, dan e. Sisi-sisi pada potongan D(b) adalah b, p, dan q.

Ada hubungan erat antara konsep pohon merentang, siklus, dan irisan, dan ini adalah isi dari dua teorema berikutnya.



Gambar 8.1 Konsep pohon merentang, siklus, dan irisan

TEOREMA 8.1.2

Misalkan T adalah pohon merentang dalam graf terhubung G, dan misalkan C dan D masing-masing adalah sikel dan cutset di G. Maka:

(a) Tidak ada persamaan sisi antara C dan D atau ada banyak sisi yang sama di antara keduanya.
(b) Sekurang-kurangnya satu sisi C adalah tali busur dari T.
(c) Setidaknya satu sisi D adalah sisi T.

Bukti:

(a) Misalkan D = (W, W′). Jika semua simpul dari C berada di salah satu dari dua himpunan bagian, maka tentu saja C dan D tidak memiliki sisi yang sama. Misalkan w dan w′ adalah dua simpul di C di mana w di W dan w′ di W. Maka siklus C yang dimulai dari w dan berakhir di w tentu akan menggunakan sisi-sisi dari D beberapa kali: tepi dari D bentuk {i, j}, di mana i di W dan j di W′, digunakan, tepi bentuk {u, v} juga digunakan, di mana u di W′ dan v adalah dalam w.
(b) Jika tidak ada rusuk dari C yang merupakan tali busur dari T, maka C adalah subgraf dari T, yang merupakan kontradiksi karena T adalah asiklik.

(c) Jika tidak ada sisi D yang merupakan sisi dari T, penghapusan sisi dari D tidak akan memutuskan G karena penghapusan tersebut tidak akan mempengaruhi pohon merentang.

TEOREMA 8.1.3

(a) Misalkan D(e) adalah cutset fundamental terhadap sisi e dari pohon merentang T, dan misalkan f adalah elemen lain dalam cutset ini. Maka (1) f adalah akord T yang mendefinisikan siklus fundamental C(f), (2) e adalah elemen C(f), dan (3) jika e' adalah akord T lain yang tidak ada di D(e), maka e bukan elemen dari C(e') .

(b) Misalkan C(e) adalah siklus fundamental terhadap akord e dari pohon merentang T, dan misalkan f adalah rusuk lainnya dalam siklus ini. Maka (1) f adalah sisi dari pohon yang mendefinisikan cutset dasar D(f), (2) e adalah elemen dari D(f), dan (3) jika e' adalah sisi lain dari T yang tidak ada di C (e), maka e bukan elemen dari D(e') .

Bukti (a):

(1) Setiap sisi f di D(e)selain e, tidak dapat menjadi sisi di T karena T adalah asiklik. Jadi f adalah akord yang mendefinisikan siklus fundamental C(f)

(2) Misalkan f adalah sembarang sisi di D(e) selain f. Maka D(e) = {e, f} A, di mana A adalah himpunan akord dari T, dan C(f) = {f}B, di mana B adalah himpunan sisi dari T. Sisi f adalah umum untuk D(e) dan C(f). Ingat bahwa D(e) dan C(f) harus memiliki jumlah sisi yang genap. Karena A dan B tidak memiliki sisi yang sama, kami menyimpulkan bahwa e adalah satu-satunya sisi lain yang sama untuk D(e) dan C(f). Jadi e adalah sisi dalam siklus fundamental C(f) .

(3) Misalkan C(e') adalah siklus fundamental terhadap e' yang merupakan akord yang tidak ada di D(e). Jadi C(e') = e' L, di mana L adalah himpunan sisi dari T, dan D(e) = e M, di mana M adalah himpunan tali dari T. Jika e dalam C(e'), maka e' akan berada di D(e), yang bertentangan dengan asumsi kami. Jadi e tidak ada di C(e').

Bukti (b):

Bukti ini mirip dengan (a) dan dibiarkan sebagai latihan.

Kita dapat menyatakan kembali bagian (a) dan (b) dari Teorema 8.1.3 sebagai berikut:

(a) Jika e adalah sembarang sisi dari pohon merentang T pada graf sederhana terhubung G, maka (1) terdapat sebuah cutset unik D(e); (2) jika f adalah sembarang sisi dalam potongan ini selain e, maka f adalah akord dari T yang mendefinisikan sebuah siklus unik C(f) sedemikian rupa sehingga e adalah sisi dalam siklus ini; dan (3) jika e' adalah tali busur dari T yang tidak berada di D(e) yang mendefinisikan sebuah siklus unik C(e') maka e bukan merupakan sisi dalam C(e').

(b) Jika e adalah sembarang tali busur dari pohon merentang T pada graf sederhana terhubung G, maka (1) terdapat siklus unik C(e); (2) jika f adalah sembarang edge dalam siklus ini selain e, maka f adalah edge dari T yang mendefinisikan sebuah cutset unik D(f) sehingga e adalah edge dalam cutset ini; dan (3) jika e' adalah edge dari T yang tidak ada di C(e) yang mendefinisikan sebuah cutset unik D(e'), maka e bukan edge di D(e') .

Jika kita mengasosiasikan bilangan real (disebut bobot) dengan setiap sisi dari graf G sehingga G menjadi jaringan, bobot pohon merentang T di G adalah jumlah bobot semua sisi di T. Rentang A pohon T adalah pohon merentang minimal (MST) jika bobot T tidak melebihi bobot pohon merentang lainnya di G. Masalah MST (juga dikenal sebagai masalah konektor minimal) adalah masalah menemukan MST dalam sebuah koneksi grafik G. Masalah optimasi ini memiliki beberapa aplikasi praktis yang penting. Sebagai contoh, akan sangat membantu dalam merencanakan jaringan komunikasi dan distribusi skala besar ketika pertimbangan yang paling penting biasanya adalah menyediakan jalur antara setiap pasangan simpul dengan cara yang paling ekonomis. Simpulnya adalah kota, terminal, atau gerai ritel, dan ujungnya adalah jalan raya atau jalur pipa. Bobot yang sesuai dengan tepi ini dapat berupa jarak atau biaya atau waktu yang terlibat dalam proses ini. Lihat Graham and Hell (1982) untuk survei lengkap dan banyak referensi. Dalam bab ini kami menyajikan dua algoritma sederhana untuk menemukan pohon merentang minimal dalam grafik. Salah satunya karena Kruskal (1956), dan yang lainnya karena Prim (1957). Pendekatan dalam kedua metode tersebut adalah serakah: Kebetulan jika kita mengambil "potongan pilihan" di setiap kesempatan tanpa melanggar aturan apa pun, pada akhirnya kita akan mendapatkan solusi yang optimal.

Masalah pohon merentang minimal memiliki generalisasi berikut. Misalkan W adalah himpunan bagian dari himpunan V dari semua simpul dari graf sederhana terhubung G. Sebuah pohon T = (U, F) di G, di mana W adalah himpunan bagian dari U, disebut pohon Steiner terhadap himpunan W Masalah jaringan Steiner minimal untuk himpunan W adalah masalah menemukan pohon Steiner terhadap W dengan bobot minimum. Jadi pohon Steiner minimal terhadap V adalah pohon merentang minimal di G. Sangat mungkin bahwa pohon Steiner minimal sehubungan dengan subset W yang tepat adalah pohon merentang minimal di G. Tidak ada algoritma efisien yang diketahui untuk menyelesaikan masalah tersebut. Masalah pohon steiner. Sebuah algoritma yang efisien untuk mendapatkan solusi perkiraan disajikan dalam Chang (1972).

8.2 ALGORITMA KRUSKAL'S GREEDY

Kami membuat daftar tepi jaringan yang terhubung dengan n simpul dalam urutan menaik dari bobot yang tidak berkurang dan kemudian membangun subgraf T yang memeriksa tepi ini satu per satu dimulai dengan tepi dengan bobot terkecil. Sebuah tepi akan ditambahkan ke T selama tidak membentuk siklus dengan beberapa atau semua tepi T. Konstruksi berhenti ketika T memiliki (n – 1) tepi. Jelas, prosedur serakah ini memastikan bahwa T adalah pohon merentang. Bahwa T yang diperoleh memang merupakan pohon merentang minimum adalah konsekuensi dari hasil berikut.

TEOREMA 7.2.1

Jika e adalah sisi dalam siklus C dari graf terhubung G sedemikian rupa sehingga bobot e lebih besar dari bobot sisi lainnya dalam siklus C, maka e bukan sisi untuk MST mana pun di G.

Bukti:

Misalkan T adalah MST di mana e adalah tepi. Misalkan D(e) adalah cutset fundamental terhadap e. Karena edge e sama untuk siklus C dan cutset D(e), harus ada setidaknya satu elemen lagi f yang sama untuk kedua set ini karena jumlah elemen yang sama untuk sebuah cycle dan cutset adalah genap. Karena f ada di D(e), f tentu saja merupakan tali busur dari T.

Misalkan C(f) adalah siklus fundamental terhadap f. Dengan Teorema 7.1.2 kita tahu bahwa e adalah elemen dari C(f). Sekarang perhatikan subgraf H yang diperoleh dengan menghubungkan f ke T. Satu-satunya siklus dalam T adalah C(f), dan jika kita menghapus e dari H, kita mendapatkan pohon rentang T′ dengan bobot kurang dari T. Kontradiksi ini menetapkan fakta bahwa e bukan tepi dari T .

Akibat Wajar:

Jika bobot sisi lain di C tidak melebihi bobot e, ada pohon merentang minimal di mana e bukan sisi.

Dalam algoritme Kruskal, kami meninggalkan tepi p demi tepi lain q untuk dimasukkan dalam T (ketika bobot p tidak melebihi q) hanya ketika penyertaan p menciptakan siklus di mana p adalah tepi dengan bobot terbesar . Dengan demikian, algoritma Kruskal menyelesaikan masalah MST dengan benar. Ini juga merupakan latihan yang mudah pada tahap ini untuk memverifikasi bahwa jika semua bobot tepi di G berbeda, ada MST unik di G.

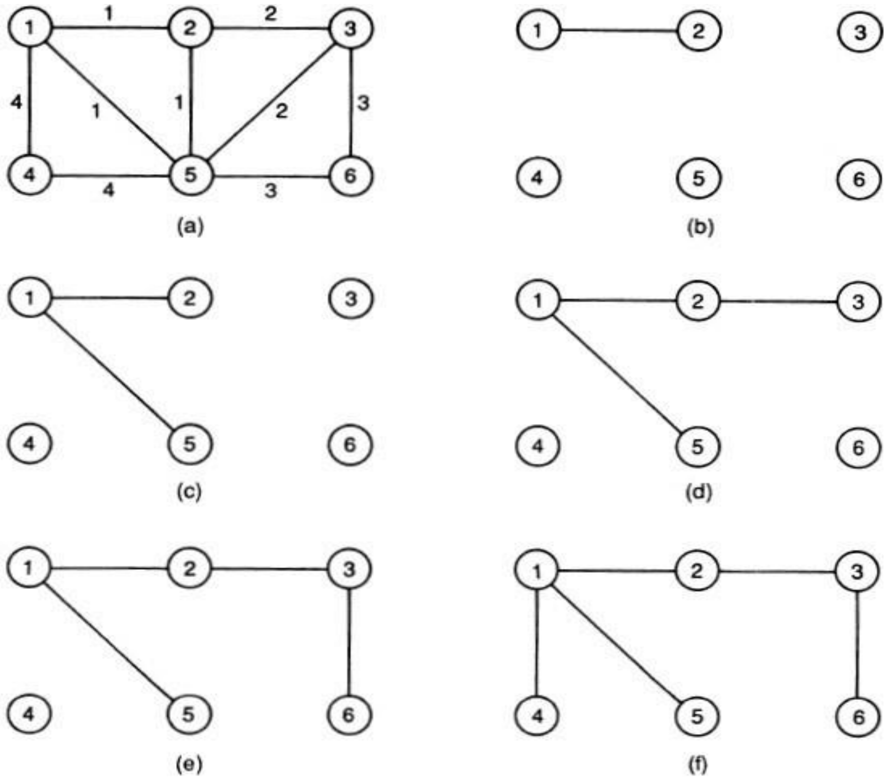
Sebagai contoh, perhatikan grafik Gambar 8.2, di mana kita memiliki daftar L yang diurutkan dari semua sisi G sebagai:

$$L = \{\{1, 2\}, (1, 5\} \{2, 5\}, \{2, 3\}, \{3, 5\}, \{3, 6\}, \{5, 6\}, \{1, 4\}, \{4, 5\}\}$$

dalam urutan menaik. Algoritme memeriksa {1, 2} dan menerimanya untuk pohon. Kemudian memeriksa {1, 5} dan menerimanya. Setelah itu memeriksa {2, 5} dan tidak menerimanya, untuk menghindari siklus C yang terdiri dari {1, 2}, {1, 5} dan {2, 5}. Kemudian ia melanjutkan lebih jauh dan menerima {2, 3}, {3, 6}, dan {1, 4} secara bergantian. Pada tahap ini berhenti karena jumlah sisi yang diterima kurang satu dari jumlah total simpul. Dalam grafik ini kami memiliki sembilan sisi dan kami harus memeriksa delapan di antaranya sebelum kami berhenti.

Berapa banyak langkah komputasi (dalam hal ini, perbandingan) yang diperlukan untuk mengatur (mengurutkan) sisi m dari grafik dengan urutan yang tidak menurun? Jumlah perbandingan tersebut tidak diragukan lagi tergantung pada algoritma yang kita gunakan untuk mengurutkan m elemen dari himpunan tepi E. Salah satu metode yang jelas adalah sebagai berikut: Secara berurutan membandingkan suku ke-i dengan suku (i + 1) dalam himpunan, menukar keduanya jika suku ke-i lebih besar dari suku ke-(i + 1). Prosedur ini disebut bubblesort karena angka yang lebih besar “naik” ke atas. Angka pertama dalam himpunan harus dibandingkan dengan paling banyak (m – 1) angka. Kemudian angka kedua harus dibandingkan dengan paling banyak (m – 2) angka, dan seterusnya. Dengan demikian jumlah total perbandingan pada kasus terburuk jika kita menggunakan algoritma bubblesort adalah 1 + 2 + 3 + . . . + (m – 1) = m(m – l)/2, yang merupakan polinomial dalam m derajat 2. Dengan kata lain kompleksitas kasus terburuk dari algoritma bubblesort adalah O(m2).

Lihat Lampiran untuk notasi dan konsep yang terkait dengan kompleksitas komputasi algoritma. Di sisi lain jika kita menggunakan algoritma mergesort (lihat Stanat dan McAllister, 1977) jumlah komputasi untuk mengurutkan m angka dalam kasus terburuk adalah O(m log m). Secara umum, tidak ada algoritma pengurutan yang lebih efisien dari ini. Algoritma terkenal lainnya yang dikenal sebagai heapsort memiliki perilaku kasus terburuk O(m log m), sedangkan algoritma quicksort memiliki perilaku kasus rata-rata O(m log m) dan perilaku kasus terburuk O(n2) , di mana n adalah jumlah simpul. Lihat Aho dkk. (1983) untuk lebih jelasnya. Dalam keadaan ini, masuk akal untuk menyimpulkan bahwa kompleksitas kasus terburuk dari algoritma Kruskal adalah O(m log m). Perhatikan, bagaimanapun, bahwa jika m sangat besar dibandingkan dengan n [yaitu, ketika m adalah O(n2)], tidak terlalu ekonomis untuk mengurutkan semua sisi m ketika kita hanya membutuhkan (n – 1) dari m sisi ini . Lihat Syslo dkk. (1983) untuk rincian implementasi algoritma Kruskal ketika m besar.



Gambar 8.2 Konsep komplesitas algoritma

8.3 ALGORITMA PRIM'S GREEDY

Dalam prosedur ini kita mulai dengan simpul sembarang v di G dan memeriksa semua sisi yang datang di v. Misalkan e = {v, w} adalah sisi dengan bobot paling kecil di antara semua sisi ini. Kami membangun subgraf T dari G dimulai dengan e sebagai tepi. Selanjutnya periksa semua sisi (selain e) yang datang di v dan semua sisi yang datang di w dan pilih satu sisi f yang bobotnya paling kecil di antara mereka. Tepi yang baru ditemukan ini ditambahkan ke subgraf T. Tepi f adalah antara v dan simpul baru atau antara w dan simpul baru. Biarkan Anda menjadi simpul baru. Pada tahap ini kita memiliki tiga simpul v, w, dan u. Periksa semua sisi selain e dan f yang datang di v, u, dan w dan pilih salah satu dengan bobot terkecil sehingga sisi e dan f dan tepi yang baru dipilih g tidak membentuk sebuah siklus. Pada tahap ini g ditambahkan ke T. Kami melanjutkan sampai semua simpul diperhitungkan. Prosedur ini jelas berakhir dengan pohon merentang. Bahwa pohon ini memang MST adalah konsekuensi dari dua akibat wajar dari teorema berikut.

TEOREMA 8.3.1

Jika v adalah sembarang simpul dalam jaringan G yang terhubung dan jika e adalah kejadian sisi di v sedemikian rupa sehingga bobot e lebih kecil dari bobot setiap kejadian sisi di v, maka e adalah sisi dari setiap pohon merentang minimum di G.

Bukti:

Misalkan T adalah MST dan misalkan e = {v, w} bukan sisi dari T. Misalkan H adalah subgraf dari G yang diperoleh dengan menambahkan e ke T. H memiliki siklus unik C(e) yang dapat direpresentasikan sebagai v- - - - v1 - - - - v2- - - - . . . . vr w v, di mana e = {v, w} dan misalkan f = {v, v1}. Sekarang baik e dan f datang di v dan bobot e lebih kecil dari pada f. Jika kita menghilangkan f H kita mendapatkan pohon merentang T' dengan bobot lebih kecil dari bobot T, yang merupakan kontradiksi.

Akibat Wajar 1

Jika v adalah sembarang simpul dari G dan jika e adalah kejadian sisi di v sedemikian rupa sehingga bobot tidak ada sisi yang datang di v lebih kecil dari bobot e, ada MST di G dimana e adalah sisi.

Akibat Wajar 2

Jika sebuah pohon T' yang merentang simpul-simpul dalam subset W dari simpul-simpul dalam graf terhubung G = (V, E) adalah subpohon dari pohon merentang minimal G, ada pohon merentang minimal G yang berisi T' dan tepi terkecil yang menghubungkan W dan V – W.

Pada setiap iterasi dari algoritma Prim kita memiliki partisi dari himpunan simpul V = {1, 2, , n} menjadi himpunan bagian P dan Q dimana P adalah himpunan simpul yang telah diperhitungkan dan Q adalah komplemennya. Awalnya, kami mengambil P = {1}. Kami mengasosiasikan label t(i) untuk setiap simpul i di Q. Awalnya, t(i) adalah bobot sisi antara 1 dan i jika ada sisi; jika tidak, itu adalah tak terhingga (angka positif yang besar). Pada langkah 1 kita memilih sebuah simpul v di Q dengan label terkecil. Kemudian kita mencari titik u di P sedemikian rupa sehingga bobot d(u, v) dari sisi antara u dan v adalah t(v). Pada titik ini tepi {u, v} diterima sebagai tepi untuk MST dan v ditambahkan ke P. Pada langkah 2 kami memperbarui label dari simpul yang tersisa di Q sebagai berikut. Jika w di Q, tentukan t(w) := Min{t(w), d(v, w)}, di mana v adalah entri terbaru di P. Kami melanjutkan dengan cara yang sama sampai P = V.

Kompleksitas kasus terburuk dari algoritma dapat diperoleh dengan segera. Awalnya, Q memiliki (n – 1) elemen. Jadi pada langkah 1, akan ada paling banyak (n – 2) perbandingan untuk memulai. Jadi langkah ini memerlukan (n – 2) + (n – 3) + . . . + 2 + 1 perbandingan. Pada langkah 2 kita memiliki (n – 2) elemen untuk memulai. Label setiap simpul w di Q harus dibandingkan dengan d(v, w), di mana v adalah entri terakhir di P. Ini melibatkan (n – 2) perbandingan untuk memulai dan kemudian (n – 3) perbandingan. Jadi, langkah 2 juga memerlukan perbandingan kasus terburuk seperti pada langkah 1. Jadi, kompleksitas kasus terburuk dari algoritma Prim adalah dua kali jumlah bilangan asli (n – 2) pertama, yaitu O(n2). Iterasi yang berbeda dari algoritma ini untuk jaringan Gambar 8.3 adalah sebagai berikut.

Iterasi 1

Langkah 1:

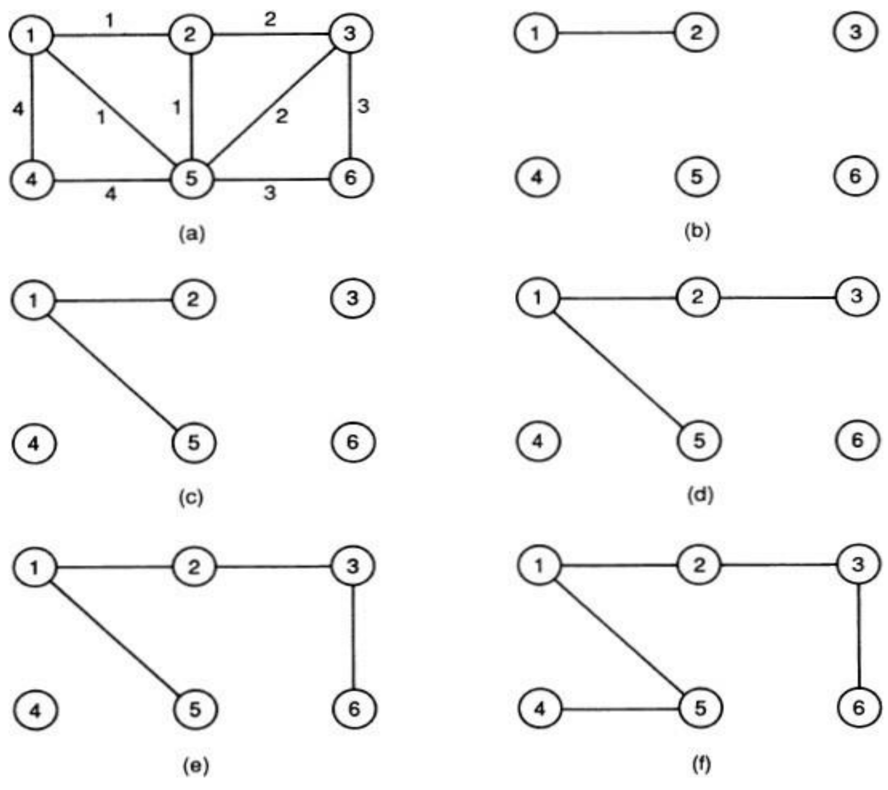
$$P = \{1\} \qquad Q = \{2, 3, 4, 5, 6\}$$

$$\begin{cases} t(2) = 1 \\ t(3) = - \\ t(4) = 4 \\ t(5) = 1 \\ t(6) = - \end{cases}$$

Label terkecil sesuai dengan simpul 2. Jadi tepi {1, 2} ada di T. Himpunan P dimutakhirkan menjadi P = {1, 2}.
Langkah 2:

$$P = \{1, 2\} \qquad Q = \{3, 4, 5, 6\}$$

$$\begin{cases} t(3) = \text{Min}\,\{t(3), d(2, 3)\} = 2 \\ t(4) = \text{Min}\,\{t(4), d(2, 4)\} = 4 \\ t(5) = \text{Min}\,\{t(5), d(2, 5)\} = 1 \\ t(6) = \text{Min}\,\{t(6), d(2, 6)\} = - \end{cases}$$



Gambar 8.3 Himpunan P dimutakhirkan

Iterasi 2
Langkah 1:

$$P = \{1, 2\} \qquad Q = \{3, 4, 5, 6\}$$

$$\begin{cases} t(3) = 2 \\ t(4) = 4 \\ t(5) = 1 \\ t(6) = - \end{cases}$$

Titik 5 dipilih untuk memperbarui P. Karena d(1, 5) = d(2, 5), kita dapat mengambil salah satu dari sisi {1, 5} atau sisi {2, 5} untuk memperbarui T.
Langkah 2:

$$P = \{1, 2, 5\} \qquad Q = \{3, 4, 6\}$$

$$\begin{cases} t(3) = \text{Min}\,\{t(3), d(5, 3)\} = 2 \\ t(4) = \text{Min}\,\{t(4), d(5, 4)\} = 4 \\ t(6) = \text{Min}\,\{t(6), d(5, 6)\} = 3 \end{cases}$$

Iterasi 3
Langkah 1:

$$P = \{1, 2, 5\} \qquad Q = \{3, 4, 6\}$$

$$\begin{cases} t(3) = 2 \\ t(4) = 4 \\ t(6) = 3 \end{cases}$$

Vertex 3 dipilih untuk memperbarui P dan tepi {2, 3} dipilih untuk memperbarui T.
Langkah 2:

$$P = \{1, 2, 5, 3\} \qquad Q = \{4, 6\}$$

$$\begin{cases} t(4) = \text{Min}\,\{t(4), d(3, 4)\} = 4 \\ t(6) = \text{Min}\,\{t(6), d(3, 6)\} = 3 \end{cases}$$

Iterasi 4
Langkah 1:

$$P = \{1, 2, 3, 5\} \qquad Q = \{4, 6)\}$$

$$\begin{cases} t(4) = 4 \\ t(6) = 3 \end{cases}$$

Vertex 6 dipilih untuk memperbarui P dan tepi {3, 6} dipilih untuk memperbarui T.

Langkah 2:

$$P = \{1, 2, 3, 5, 6\} \qquad Q = \{4\}$$

$$t(4) = \text{Min}\{t(4), d(6, 4)\} = 4$$

Iterasi 5

Langkah 1:

Vertex 4 dipilih untuk memperbarui P dan tepi {5, 4} dipilih untuk memperbarui T.

Langkah 2:

P = {1, 2, 3, 4, 5, 6} Q = himpunan kosong.

Output: Tepi M.S.T. adalah {1, 2}, {1, 5}, {2, 3}, {3, 6}, dan {4, 5}.

Algoritma Prim (Metode Matriks)

Misalkan D = (d(i, j)) adalah matriks n × n, di mana n adalah jumlah simpul dari G dan d(i, j) adalah bobot sisi {i, j} jika ada sisi antara i dan j, Jika tidak, itu tak terhingga. Awalnya kami menghapus semua elemen kolom 1 dan menandai baris 1 dengan a*. Semua elemen awalnya tanpa garis bawah. Setiap iterasi memiliki dua langkah sebagai berikut:

Langkah 1: Pilih elemen terkecil dari entri (tanpa garis bawah) di baris berbintang. Berhenti jika tidak ada elemen seperti itu. Tepi yang sesuai dengan entri yang digarisbawahi merupakan MST.

Langkah 2: Jika d(i, j) dipilih pada langkah 1, garis bawahi entri tersebut, tandai baris j dengan a *, dan hapus elemen yang tersisa di kolom j. Pergi ke langkah 1.

Sebagai ilustrasi, mari kita perhatikan jaringan seperti yang diberikan pada Gambar 8.2 Awalnya, kami memiliki:

$$D = \begin{bmatrix} - & 1 & - & 4 & 1 & - \\ - & - & 2 & - & 1 & - \\ - & 2 & - & - & 2 & 3 \\ - & - & - & - & 4 & - \\ - & 1 & 2 & 4 & - & 3 \\ - & - & 3 & - & 3 & - \end{bmatrix} \begin{array}{l} * \\ \\ \\ \\ \\ \\ \end{array}$$

di mana semua entri di kolom 1 telah dihapus dan baris 1 diberi bintang. Pada titik ini tidak ada entri yang digaris bawahi.

Iterasi 1:

$$D = \left[\begin{matrix} - & \underline{1} & - & 4 & 1 & - \\ - & - & 2 & - & 1 & - \\ - & - & - & - & 2 & 3 \\ - & - & - & - & 4 & - \\ - & - & 2 & 4 & - & 3 \\ - & - & 3 & - & 3 & - \end{matrix}\right] \begin{matrix} * \\ * \\ \\ \\ \\ \end{matrix}$$

Iterasi 2:

$$D = \left[\begin{matrix} - & \underline{1} & - & 4 & \underline{1} & - \\ - & - & 2 & - & - & - \\ - & - & - & - & - & 3 \\ - & - & - & - & - & - \\ - & - & 2 & 4 & - & 3 \\ - & - & 3 & - & - & - \end{matrix}\right] \begin{matrix} * \\ * \\ \\ \\ * \\ \end{matrix}$$

Iterasi 3:

$$D = \left[\begin{matrix} - & \underline{1} & - & 4 & \underline{1} & - \\ - & - & \underline{2} & - & - & - \\ - & - & - & - & - & 3 \\ - & - & - & - & - & - \\ - & - & - & 4 & - & 3 \\ - & - & - & - & - & - \end{matrix}\right] \begin{matrix} * \\ * \\ * \\ \\ * \\ \end{matrix}$$

Iterasi 4:

$$D = \left[\begin{matrix} - & \underline{1} & - & 4 & \underline{1} & - \\ - & - & \underline{2} & - & - & - \\ - & - & - & - & - & \underline{3} \\ - & - & - & - & - & - \\ - & - & - & 4 & - & - \\ - & - & - & - & - & - \end{matrix}\right] \begin{matrix} * \\ * \\ * \\ \\ * \\ * \end{matrix}$$

Iterasi 5:

$$D = \begin{array}{c} \begin{bmatrix} - & \underline{1} & - & \underline{4} & \underline{1} & - \\ - & - & \underline{2} & - & - & - \\ - & - & - & - & - & \underline{3} \\ - & - & - & - & - & - \\ - & - & - & - & - & - \\ - & - & - & - & - & - \end{bmatrix} \end{array} \begin{array}{c} * \\ * \\ * \\ \\ * \\ \\ \end{array}$$

Prosedur pada tahap ini berhenti, memberikan tepi yang sesuai dengan entri yang digarisbawahi dalam matriks D dari iterasi terakhir. Jadi rusuk {1, 2}, {1, 4}, {1, 5}, {2, 3}, dan {3, 6} membentuk pohon merentang minimal di G.

8.4 PERBANDINGAN DUA ALGORITMA

Waktu eksekusi algoritma Prim hanya bergantung pada jumlah simpul, tetapi waktu untuk algoritma Kruskal meningkat karena jumlah tepi meningkat untuk jaringan dengan jumlah simpul yang sama. Namun, secara umum, tidak mungkin untuk menyatakan mana yang lebih efisien. Efisiensi tergantung pada, antara lain, struktur jaringan dan distribusi bobot. Banyak variasi, terutama berdasarkan struktur data dan detail implementasi, telah disarankan untuk meningkatkan efisiensi. Telah diamati bahwa untuk jaringan dengan simpul hingga 100, metode Prim tampaknya lebih efisien, terutama ketika ada banyak tepi. Waktu berjalan berikut untuk dua algoritme yang dijalankan pada komputer AMDHL 470 V/8 dilaporkan oleh Syslo et al. (1983).

Tabel 8.1 Perbandingan dua algoritma

Number of vertices	Number of edges	Execution time (msec)	
		Kruskal	Prim
80	790	64	52
80	1580	95	51
80	3160	186	50
100	200	44	78
100	300	66	80

8.5 CATATAN DAN REFERENSI

Untuk diskusi tentang pohon merentang secara umum, lihat buku standar tentang teori graf yang tercantum di akhir buku. Lihat Lawler (1976), di mana algoritma ("bukan yang sangat bagus") dijelaskan untuk memecahkan masalah pohon Steiner. Untuk masalah MST referensi paling awal mungkin adalah makalah klasik Kruskal (1956) dan Prim (1957). Deskripsi matriks dari algoritma Prim adalah dari Hu (1982). Untuk deskripsi detail implementasi kedua algoritma serakah ini, referensi yang sangat bagus adalah buku karya Syslo et al. (1983).

Referensi umum lainnya adalah Cheriton dan Tarjan (1976), Graham and Hell (1982), dan Gabow et al. (1986).

8.6 LATIHAN

8.1 Misalkan G adalah graf di mana T adalah pohon merentang, C adalah sebuah siklus, dan D adalah sebuah cutset. Buktikan berikut ini:

(a) C dan D memiliki banyak sisi yang genap.

(b) D dan T memiliki setidaknya satu sisi yang sama.

(c) C dan komplemen dari T memiliki setidaknya satu sisi yang sama.

8.2 Buktikan bahwa jika bobot sisi-sisi pada graf terhubung semuanya berbeda, terdapat MST unik dalam graf tersebut.

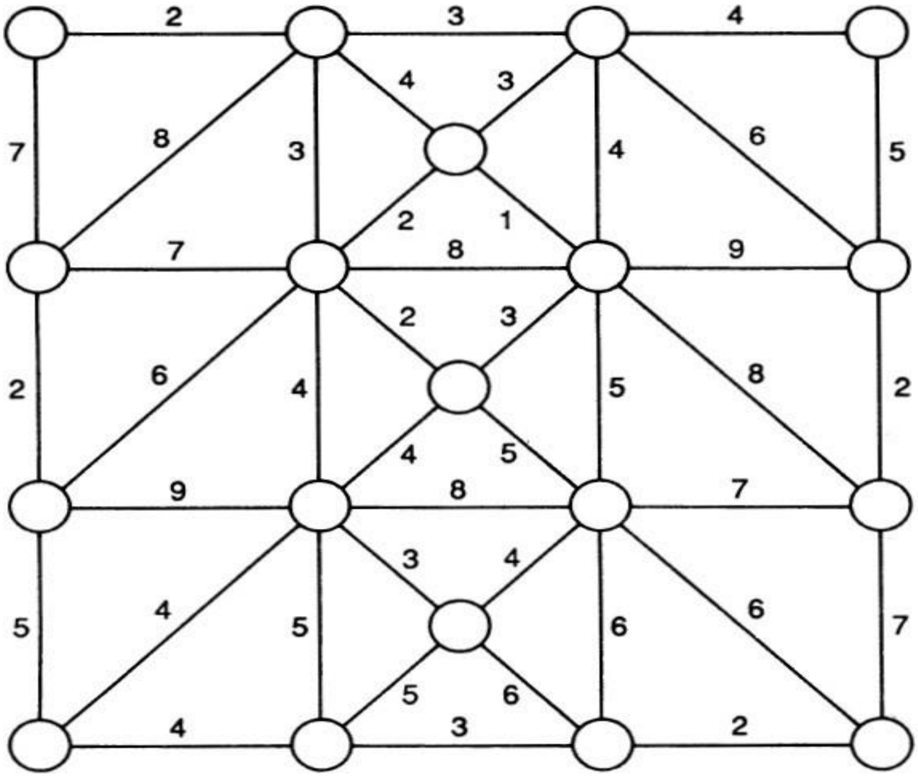
8.3 7.3. Jika e adalah edge unik pada jaringan terhubung dengan bobot terkecil, buktikan bahwa e adalah edge di setiap MST di G.

8.4 (a) Dapatkan pohon Steiner sehubungan dengan himpunan simpul W = {1, 2, 4, 5} dalam jaringan yang ditunjukkan pada Gambar 7.2.1. (b) Misalkan bobot rusuk {1, 5}, {2, 5}, dan {4, 5} masing-masing 10 satuan. Apa yang akan menjadi pohon Steiner terhadap W?

8.5 Gunakan algoritma Kruskal untuk mendapatkan MST di G dengan matriks bobot berikut:

$$\begin{bmatrix} 0 & 20 & 15 & 4 & 3 & - \\ 20 & 0 & 19 & - & - & 9 \\ 15 & 19 & 0 & 8 & - & 10 \\ 4 & - & 8 & 0 & 6 & 9 \\ 3 & - & - & 6 & 0 & 7 \\ - & 9 & 10 & 9 & 7 & 0 \end{bmatrix}$$

8.6 Gunakan algoritma Prim (metode matriks) untuk mendapatkan MST pada grafik Soal 8.5. Apakah Anda mendapatkan pohon yang sama di Soal 8.5 dan Soal 8.6? Tentukan berat kedua pohon tersebut.



Gambar 8.4 Menggunakan algoritma Prim untuk mendapatkan MST

8.7 Modifikasi algoritma Kruskal untuk mendapatkan pohon merentang maksimum.

8.8 Dapatkan pohon merentang maksimum dalam grafik Soal 8.5.

8.9 Hapus sebanyak mungkin sisi dari graf G pada Gambar 8.4 untuk mendapatkan graf terhubung G′ sedemikian rupa sehingga bobot G′ minimum.

8.10 Siklus dalam jaringan terhubung G yang melewati setiap simpul G disebut siklus Hamilton. Sebuah jaringan terhubung arbitrer G tidak perlu memiliki siklus Hamilton. Jumlah bobot semua sisi dalam siklus Hamilton adalah bobot siklus Hamilton, dan siklus Hamilton dengan bobot minimum disebut siklus travelling salesman (TS). Tunjukkan bahwa adalah mungkin untuk memperoleh batas bawah untuk bobot siklus TS dalam jaringan terhubung G menggunakan algoritma MST baik G memiliki siklus Hamilton atau tidak. Dapatkan ikatan seperti itu untuk jaringan dalam Soal 8.10.

8.11 Bangun jaringan dengan 5 simpul dan 5 tepi dengan bobot berbeda di mana MST uniknya adalah pohon Steiner minimal terhadap 4 simpul ini.

BAB 9
MASALAH JALUR TERPENDEK

9.1 PENDAHULUAN

Jika setiap busur digraf diberi bobot numerik (yaitu, jarak), itu adalah masalah alami dan menarik secara intuitif untuk menemukan jalur terpendek (jika ada) dari simpul yang ditentukan ke simpul lain yang ditentukan. Banyak masalah optimasi dapat dirumuskan dan diselesaikan sebagai masalah jalur terpendek dari jenis ini, dan banyak masalah kompleks dalam riset operasi dapat diselesaikan dengan prosedur yang memanggil algoritma jalur terpendek sebagai subrutin. Masalah jalur terpendek sebenarnya adalah masalah yang paling mendasar dan juga yang paling sering ditemui dalam optimasi kombinatorial. Menurut Goldman (1982), algoritma jalur terpendek yang dikembangkan oleh Departemen Perhubungan AS secara teratur digunakan miliaran kali setiap tahun. Kami membatasi perhatian kami pada dua jenis masalah: (1) masalah menemukan jalur terpendek dari simpul v ke simpul lain w, dan (2) masalah menemukan jalur terpendek dari setiap simpul ke setiap simpul lainnya. Tentu saja, (1) adalah kasus khusus dari (2).

Berikut ini akan dibahas dua algoritma polinomial untuk menyelesaikan masalah jalur terpendek (S.P.). Algoritma pertama adalah mencari S.P. dan jarak terpendek (S.D.) dari sebuah simpul tertentu ke setiap simpul lainnya. Algoritma ini disebabkan oleh Dijkstra (1959). Algoritme kami berikutnya, yang dikenal sebagai algoritme Floyd-Warshall, memungkinkan kami menemukan S.P. dan S.D. dari setiap simpul ke setiap simpul lainnya. Prosedur ini disebabkan oleh Floyd (1964) dan Warshal (1962). Kami berasumsi bahwa fungsi bobot adalah non-negatif dalam kasus algoritma Dijkstra meskipun pembatasan ini dimungkinkan untuk dilonggarkan. Perlu dicatat bahwa ada perbedaan nyata antara masalah yang melibatkan fungsi bobot non-negatif dan masalah yang melibatkan fungsi bobot arbitrer. Dalam kasus terakhir, masalahnya menjadi tidak terbatas jika jaringan memiliki siklus dengan bobot negatif. Algoritma Floyd-Warshall mendeteksi adanya siklus negatif tersebut.

9.2 ALGORITMA DIJKSTRA

Dalam jaringan G = (V, E), misalkan V = {1, 2, . . . , n} dan misalkan bobot busur (i, j) adalah a(i, j), yang diasumsikan nonnegatif. Jika tidak ada busur dari i ke j (i j), maka a(i, j) diambil tak hingga. Dengan demikian, kita memiliki matriks bobot $n \times n$ A = (a(i, j)) untuk G, di mana semua bilangan diagonalnya adalah 0. Masalahnya adalah menemukan S.D. dan S.P. dari simpul 1 ke semua simpul lainnya.

Prosedurnya adalah sebagai berikut. Setiap simpul i diberi label yang bersifat permanen atau tentatif. Label permanen L(i) dari i adalah S.D. dari 1 ke i, sedangkan label tentatif L'(i) dari i adalah batas atas S.D. dari 1 sampai i. Pada setiap tahap prosedur, P adalah himpunan simpul dengan label permanen dan T adalah komplemennya. Awalnya, P = {1} dengan L(1) = 0 dan L'(i) = a(1, i) untuk setiap i. Ketika P = V algoritma berhenti. Setiap iterasi pada dasarnya terdiri dari dua langkah, sebagai berikut:

Langkah 1 (Penunjukan Label Permanen):
Temukan simpul k di T yang L'(k) minimal. Berhenti jika tidak ada k seperti itu karena tidak ada jalur dari 1 ke sembarang titik di T. Dekatkan k ke himpunan P. Berhenti jika P = V.
Langkah 2 (Revisi Label Tentatif):
Jika j adalah simpul di T, ganti L'(j) dengan nilai yang lebih kecil dari L'(j) dan L(k) + a(k, j) . Pergi ke langkah 1.

Kami sekarang membuktikan bahwa algoritma memecahkan masalah dengan benar dengan induksi pada jumlah elemen di P.

TEOREMA 9.2.1
Algoritma Dijkstra menemukan S.D. dari 1 ke masing-masing i.
Bukti:
Kami membuktikan dengan induksi pada jumlah elemen di P bahwa untuk setiap i di P, L(i) sama dengan S.D. dari 1 ke i, dan untuk setiap j tidak di P, L'(j) adalah panjang S.P. dari 1 ke j, setiap simpul perantaranya ada di P. Ini benar jika P memiliki satu elemen. Misalkan ini benar ketika P memiliki hingga m elemen. Dengan hipotesis induksi tepat sebelum simpul k disatukan dengan himpunan P, L'(k) sama dengan panjang S.P. dari 1 sampai k di mana setiap simpul antara adalah simpul di P. Sekarang k berbatasan dengan P dan L (k) = L'(k) . Kami mengklaim bahwa L(k) adalah S.D. dari 1 sampai k Jika tidak, biarkan d menjadi S.D. dari 1 sampai k Maka d < L(k) = L'(k) . Jadi setiap S.P. dari 1 hingga k harus memiliki setidaknya satu simpul bukan dari P sebagai simpul perantara. Biarkan v menjadi simpul pertama seperti itu. Biarkan aku menjadi S.D. dari 1 sampai v. Maka, jelas, d' d. Tetapi d < L'(k), yang menyiratkan bahwa d < L'(k) . Ini bertentangan dengan asumsi bahwa L'(k) minimal.

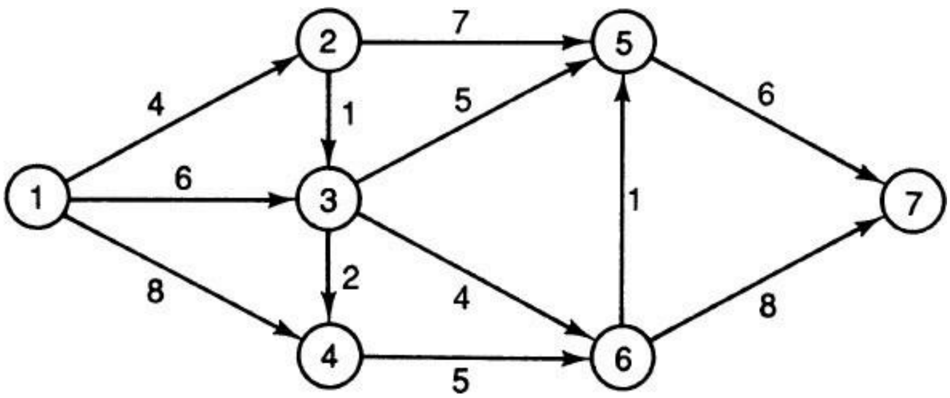
Kompleksitas kasus terburuk dari algoritma adalah O(n2) . Hal ini dapat ditetapkan sebagai berikut. Paling banyak ada n iterasi. Pada Langkah 1, pada iterasi pertama paling banyak (n – 2) perbandingan, pada iterasi berikutnya paling banyak (n – 3) perbandingan, dan seterusnya. Oleh karena itu, akan ada paling banyak (n – 2) + (n – 3) + . . . + 1 perbandingan pada Langkah 1. Pada Langkah 2 kita kembali memiliki paling banyak (n – 2) perbandingan dan juga (n – 2) penambahan pada iterasi pertama. Jadi pada Langkah 2 kita memiliki (n – 2) + (n – 3) + . . . + 1 perbandingan dan jumlah penambahan yang sama dalam kasus terburuk. Jadi dalam semua kita memiliki (n – 1)(n – 2) perbandingan dan (n – l)(n – 2)/2 jumlah penambahan, yang menetapkan kompleksitas polinomial dari algoritma.

Setelah S.D. dari 1 ke i diketahui mudah untuk menemukan S.P. dari 1 ke i dengan memeriksa simpul j sedemikian rupa sehingga (1) L(j) lebih kecil dari L(i) dan (2) terdapat busur dari j ke i. Berikut adalah contoh ilustratif untuk menemukan S.D. dan S.P. dari simpul 1 ke simpul yang tersisa dalam jaringan terarah seperti yang ditunjukkan pada Gambar 9.1.

Iterasi 1
Langkah 1:

$$P = \{1\} \qquad T = \{2, 3, 4, 5, 6, 7\}$$

$$L(1) = 0 \qquad \begin{cases} L'(2) = 4 \\ L'(3) = 6 \\ L'(4) = 8 \\ L'(5) = - \\ L'(6) = - \\ L'(7) = - \end{cases}$$



Gambar 9.1 Ilustrasi simpul 1 ke simpul tersisa dalam jaringan searah

Vertex 2 mendapat label permanen.
Langkah 2:

$$P = \{1, 2\} \qquad T = T - \{2\} = \{3, 4, 5, 6, 7\}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \end{cases} \qquad \begin{cases} L'(3) = \text{Min}\,\{6,\ L'(2) + a(2, 3)\} \\ L'(4) = \text{Min}\,\{8,\ L'(2) + a(2, 4)\} \\ L'(5) = \text{Min}\,\{-, L'(2) + a(2, 5)\} \\ L'(6) = \text{Min}\,\{-, L'(2) + -\} \\ L'(7) = \text{Min}\,\{-, L'(2) + -\} \end{cases}$$

Iterasi 2
Langkah 1:

$$P = \{1, 2\} \qquad T = \{3, 4, 5, 6, 7\}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \end{cases} \qquad \begin{cases} L'(3) = 5 \\ L'(4) = 8 \\ L'(5) = 11 \\ L'(6) = - \\ L'(6) = - \end{cases}$$

Vertex 3 mendapat label permanen.
Langkah 2:

$$P = \{1, 2, 3\} \qquad T = T - \{3\} = \{4, 5, 6, 7\}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \end{cases} \qquad \begin{cases} L'(4) = \text{Min}\,\{8, L'(3) + a(3, 4)\} \\ L'(5) = \text{Min}\,\{11, L'(3) + a(3, 5)\} \\ L'(6) = \text{Min}\,\{-, L'(3) + a(3, 6)\} \\ L'(7) = \text{Min}\,\{-, L'(3) + -\} \end{cases}$$

Iterasi 3

Langkah 1:

$$P = \{1, 2, 3\} \qquad T = \{4, 5, 6, 7\}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \end{cases} \qquad \begin{cases} L'(4) = 7 \\ L'(5) = 10 \\ L'(6) = 9 \\ L'(7) = - \end{cases}$$

Vertex 4 mendapat label permanen.

Langkah 2:

$$P = \{1, 2, 3, 4\} \qquad T = T - \{4\} = \{5, 6, 7\}$$

$$\begin{matrix} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \\ L(4) = 7 \end{matrix} \qquad \begin{matrix} L'(5) = \text{Min}\,\{10, L'(4) + -\} \\ L'(6) = \text{Min}\,\{9, L'(4) + d(4, 6)\} \\ L'(7) = \text{Min}\,\{-, L'(4) + -\} \end{matrix}$$

Iterasi 4

Langkah 1:

$$P = \{1, 2, 3, 4\} \qquad T = \{5, 6, 7\}$$

$$\begin{matrix} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \\ L(4) = 7 \end{matrix} \qquad \begin{matrix} L'(5) = 10 \\ L'(6) = 9 \\ L'(7) = - \end{matrix}$$

Vertex 6 mendapat label permanen.

Langkah 2:

$$P = \{1, 2, 3, 4, 6\} \qquad T = \{5, 6, 7\} - \{6\} = \{5, 7\}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \\ L(4) = 7 \\ L(6) = 9 \end{cases} \qquad \begin{cases} L'(5) = \text{Min}\,\{10, L'(6) + d(6, 5)\} \\ L'(7) = \text{Min}\,\{-, L'(6) + d(6, 7)\} \end{cases}$$

Iterasi 5

Langkah 1:

$$P = \{1, 2, 3, 4, 6\} \qquad T = \{5, 7\}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \\ L(4) = 7 \\ L(6) = 9 \end{cases} \qquad \begin{cases} L'(5) = 10 \\ L'(7) = 17 \end{cases}$$

Vertex 5 mendapat label permanen.

Langkah 2:

$$P = \{1, 2, 3, 4, 6, 5\} \qquad T = \{5, 7\} - \{5\}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \\ L(4) = 7 \\ L(6) = 9 \\ L(5) = 10 \end{cases} \qquad L'(7) = \text{Min}\,\{17, L'(5) + d(5, 7)\}$$

Iterasi 6

Langkah 1:

$$P = \{1, 2, 3, 4, 5, 6\} \qquad T = \{7\}$$
$$L'(7) = 16$$

Vertex 7 mendapat label permanen.

Langkah 2:

$$P = \{1, 2, 3, 4, 5, 6, 7\} \qquad T = \text{the empty set.}$$

$$\begin{cases} L(1) = 0 \\ L(2) = 4 \\ L(3) = 5 \\ L(4) = 7 \\ L(5) = 10 \\ L(6) = 9 \\ L(7) = 16 \end{cases}$$

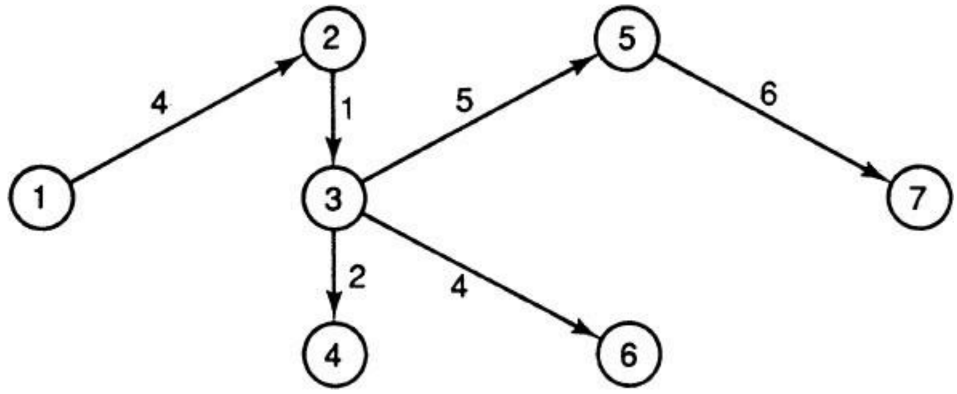
Setelah kami mendapatkan S.D. dari titik 1 ke setiap titik, sangat mudah untuk menentukan jalur terpendek dari 1 ke setiap titik. Hal ini dicapai dengan membangun pohon jarak terpendek yang berakar pada 1 sebagai berikut.

Untuk setiap simpul i (selain 1), cari simpul j sedemikian rupa sehingga (1) terdapat busur dari j ke i dalam jaringan, (2) L(j) < L(i), dan (3) L(j) + a(j, i) = L(i) . Tie-breaking adalah sewenang-wenang. Sertakan busur (j, i) di pohon. Dalam contoh kami ada busur dari 3 dan 4 hingga 6. Kami melihat bahwa:

$$L(3) + a(3, 6) = 5 + 4 = 9 = L(6)$$

dan

$$L(4) + a(4, 6) = 7 + 5 = 12$$



Gambar 9.2 Pohon berakar

Jadi busur (3, 6) ada di pohon. Sangat mudah terlihat bahwa ada ikatan antara (3, 5) dan (6, 5) untuk dimasukkan ke dalam pohon dan kita hanya dapat mengambil salah satunya. Pada Gambar 9.2 kita memiliki S.D. pohon berakar pada 1, memberikan jalur terpendek dari 1 ke semua simpul lainnya.

Catatan: Algoritma Dijkstra tidak perlu menyelesaikan S.D. masalah untuk fungsi bobot arbitrer. Pertimbangkan jaringan G = (V, E) di mana V = {1, 2, 3} dan busurnya adalah (1, 2), (1, 3), dan (2, 3) dengan bobot 10, 8, dan – 3, masing-masing. Pada iterasi 2 kita mendapatkan L(3) = 8, tetapi S.D. dari 1 sampai 3 hanya 7.

9.3 ALGORITMA FLOYD-WARSHALL

Kami melihat bahwa algoritma Dijkstra tidak cocok ketika fungsi bobotnya arbitrer. Ada beberapa algoritma polinomial untuk menyelesaikan S.D. masalah ketika fungsi bobot tidak dibatasi menjadi nonnegatif. Salah satu algoritma yang terkenal adalah algoritma Floyd-Warshall, yang dapat digunakan untuk menemukan S.D. dan S.P. dari setiap simpul ke setiap simpul lainnya untuk fungsi bobot arbitrer dan yang menempatkan keberadaan siklus negatif. Jika ada siklus negatif yang dimulai di i dan berakhir di i, tidak masuk akal untuk mempertimbangkan S.D. dari i ke setiap simpul dalam jaringan dalam masalah minimisasi.

Pertimbangkan jaringan terarah dengan n simpul dan fungsi bobot arbitrer. Misalkan A = (aij) adalah matriks bobot n × n dan misalkan P = (pij) adalah matriks n × n lainnya di mana pij = j. Kami memiliki n iterasi selama eksekusi algoritma.

Iterasi j dimulai dengan dua matriks n × n A(j–1) dan P(j–1) (awalnya A(0) = A dan P(0) = P) dan diakhiri dengan A(j) dan P(j) . Unsur-unsur dalam matriks ini didefinisikan sebagai berikut:

Jika maka entri (i, k) di A(j–l) sama dengan entri (i, k) di A(j) dan entri (i, k) di P(j–1) sama dengan (i, k)) entri di P(j) . Jika tidak, entri (i, k) di A(j) adalah jumlah entri (i, j) dan entri (j, k) di A(j-1) dan entri (i, k) di P (j) sama dengan entri (i, j) di P(j–1).

Ketika algoritma berakhir, kita memiliki dua matriks, A′ = A(n) dan P′ = P(n). Dapat dibuktikan dengan induksi bahwa entri (i,j) pada A′ adalah S.D. dari saya ke j. Lihat Papadimitriou dan Steiglitz (1982) untuk bukti. Dapat juga dibuktikan bahwa jika entri (i, j) pada P' adalah k, maka (i, k) adalah busur pertama dalam S.P. dari i ke j dan kita menggunakan fakta ini untuk memperoleh S.P. dari i ke j.

Prosedur memperbarui matriks A ini (dikenal sebagai operasi rangkap tiga) melibatkan paling banyak (n – l)2 perbandingan dan jumlah penambahan yang sama untuk setiap iterasi, sedangkan prosedur untuk memperbarui matriks P tidak melibatkan pekerjaan apa pun. Jadi kompleksitas kasus terburuk adalah O(n3), karena terdapat paling banyak n iterasi.

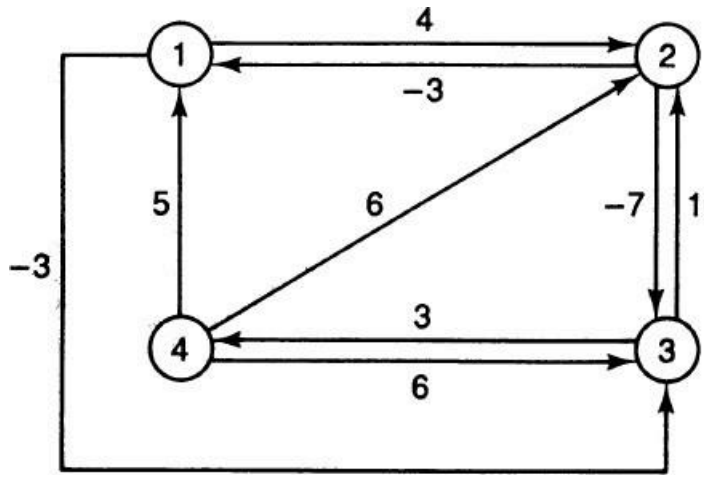
Mari kita ilustrasikan prosedur ini dalam kasus jaringan yang ditunjukkan pada Gambar 9.3. Kami menghitung matriks

$$A^{(0)} = \begin{bmatrix} 0 & 4 & -3 & - \\ -3 & 0 & -7 & - \\ - & 10 & 0 & 3 \\ 5 & 6 & 6 & 0 \end{bmatrix} \qquad P^{(0)} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{bmatrix}$$

$$A^{(1)} = \begin{bmatrix} 0 & 4 & -3 & - \\ -3 & 0 & -7 & - \\ - & 10 & 0 & 3 \\ 5 & 6 & 2 & 0 \end{bmatrix} \qquad P^{(1)} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 1 & 4 \end{bmatrix}$$

$$A^{(2)} = \begin{bmatrix} 0 & 4 & -3 & - \\ -3 & 0 & -7 & - \\ 7 & 10 & 0 & 3 \\ 3 & 6 & -1 & 0 \end{bmatrix} \qquad P^{(2)} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 2 & 2 & 3 & 4 \\ 2 & 2 & 2 & 4 \end{bmatrix}$$

$$A^{(3)} = \begin{bmatrix} 0 & 4 & -3 & 0 \\ -3 & 0 & -7 & -4 \\ 7 & 10 & 0 & 3 \\ 3 & 6 & -1 & 0 \end{bmatrix} \qquad P^{(3)} = \begin{bmatrix} 1 & 2 & 3 & 3 \\ 1 & 2 & 3 & 3 \\ 2 & 2 & 3 & 4 \\ 2 & 2 & 2 & 4 \end{bmatrix}$$



Gambar 9.3 Ilustarai prosedur kasus jaringan

dan

$$A' = A^{(4)} = \begin{bmatrix} 0 & 4 & -3 & 0 \\ -3 & 0 & -7 & -4 \\ 6 & 9 & 0 & 3 \\ 3 & 6 & -1 & 0 \end{bmatrix} \qquad P' = P^{(4)} = \begin{bmatrix} 1 & 2 & 3 & 3 \\ 1 & 2 & 3 & 3 \\ 4 & 4 & 3 & 4 \\ 2 & 2 & 2 & 4 \end{bmatrix}$$

Dari A' kita melihat bahwa S.D. dari simpul 3 ke simpul 1 adalah entri (3, 1) dalam matriks tersebut, yaitu 6. Dari P' kita melihat bahwa entri (3, 1) adalah 4. Jadi (3, 4) adalah busur pertama dalam S.P. dari 3 ke 1. Entri (4, 1) adalah 2. Jadi busur berikutnya adalah (4, 2). Kemudian kita melihat bahwa entri (2, 1) adalah 1. Jadi busur terakhir adalah (2, 1). Selanjutnya pertimbangkan digraf dengan empat simpul yang kita miliki:

$$A = \begin{bmatrix} 0 & - & - & 1 \\ 2 & 0 & 1 & - \\ - & - & 0 & - \\ - & -4 & 3 & 0 \end{bmatrix}$$

Pada akhir iterasi kedua kita memiliki:

$$A^{(2)} = \begin{bmatrix} 0 & - & - & 1 \\ 2 & 0 & 1 & 3 \\ - & - & 0 & - \\ -2 & -4 & 3 & -1 \end{bmatrix} \qquad P^{(2)} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 1 \\ 1 & 2 & 3 & 4 \\ 2 & 2 & 3 & 2 \end{bmatrix}$$

Kami melihat perkembangan baru di sini: Elemen diagonal (4, 4) adalah negatif bukannya 0, menunjukkan adanya siklus negatif dalam jaringan. Dalam matriks P yang sesuai kita melihat bahwa elemen (4, 4) adalah 2, memberikan busur (4, 2); elemen (2, 4) adalah 1, memberikan busur (2, 1), dan elemen (1, 4) adalah 4, memberikan busur (1, 4) menciptakan siklus ④- - - - - ②- - - - - -① - - - - - -④ - - - - - -dengan total berat – 1.

9.4 PERBANDINGAN DUA ALGORITMA

Untuk menyelesaikan semua pasangan (yaitu, dari setiap simpul ke setiap simpul lainnya) S.D. masalah, tampak bahwa rata-rata algoritma Dijkstra akan mengungguli Floyd-Warshall, seperti dapat dilihat dari tabel berikut di Syslo et al. (1983).
Waktu Komputasi untuk All-Pair Shortest Path Algorithms pada Jaringan Lengkap

Tabel 9.1 Perbandingan dua algoritma

Number of vertices	Run time (sec)	
	Dijkstra	Floyd–Warshall
40	0.527	0.646
60	1.767	2.156
80	4.208	5.078
100	8.052	9.862

9.5 CATATAN DAN REFERENSI

Tidak ada masalah lain dalam optimasi jaringan yang mendapat perhatian sebanyak masalah jarak terpendek. Untuk ulasan yang sangat baik, lihat Dreyfus (1969). Beberapa referensi umum yang sangat baik adalah bab-bab yang relevan dalam buku-buku oleh Lawler (1976), Minieka (1978), dan Papadimitriou dan Steiglitz (1982). Makalah oleh Dijkstra (1959) adalah salah satu makalah paling awal tentang topik ini. Lihat Nemhauser (1972) untuk perluasan algoritma Dijkstra untuk jaringan dengan bobot arbitrer. Floyd-Algoritma Warshall diterbitkan sebagai algoritma ALGOL oleh Floyd (1964) berdasarkan karya Warshall (1962). Algoritma ini sejauh ini merupakan salah satu algoritma yang dikenal paling efisien untuk memecahkan masalah jarak terpendek semua-pasangan. Untuk algoritma efisien lainnya, lihat Tabourier (1973).

9.6 LATIHAN

9.1 Matriks jarak suatu digraf adalah sebagai berikut:

$$\begin{bmatrix} 0 & - & 4 & 10 & 3 & - & - \\ - & 0 & -1 & -1 & 2 & 11 & 0 \\ - & 9 & 0 & 8 & 3 & 2 & 1 \\ - & 4 & 0 & 0 & 8 & 6 & 3 \\ - & 0 & 1 & 2 & 0 & 3 & -1 \\ - & -1 & -1 & 3 & 2 & 0 & 0 \\ - & 4 & 3 & - & - & 2 & 0 \end{bmatrix}$$

Temukan A(7) dan P(7) menggunakan algoritma Floyd–Warshall.

9.2 Temukan S.D. dan S.P. dari simpul 4 ke simpul 7 pada Soal 8.1.

9.3 Buatlah pohon berarah yang berakar pada simpul 1 dengan memberikan S.D. dari 1 ke simpul lain dalam masalah 8.1.

9.4 Ganti angka – 1 yang muncul di kolom keempat matriks pada Soal 8.1 dengan – 3. Anda mendeteksi siklus negatif sekarang. Apa siklus negatif ini?

9.5 Temukan S.P. dari 4 hingga 2 yang tidak melewati 5, 6, atau 7 pada soal 8.1.

9.6 Ganti – 1 dengan 1 dalam matriks Soal 8.1 dan temukan pohon yang berakar pada simpul 1 yang memberikan S.D. dari simpul 1 ke semua simpul menggunakan algoritma Dijkstra.

9.7 Di bandara kecil tapi berkembang, perusahaan penerbangan lokal membeli kereta traktor-trailer baru untuk membawa barang bawaan ke dan dari pesawat. Sistem bagasi mekanis baru akan dipasang dalam tiga tahun dan traktor tidak akan diperlukan lagi setelah itu. Namun, mungkin lebih ekonomis untuk mengganti traktor setelah satu atau dua tahun karena, karena penggunaan yang berat, waktu pengoperasian dan biaya perawatan akan meningkat pesat seiring bertambahnya usia. Array berikut memberikan total biaya diskon bersih yang terkait dengan pembelian traktor (harga beli dikurangi tunjangan tukar tambah ditambah biaya pengoperasian dan pemeliharaan) pada akhir tahun i dan memperdagangkannya pada akhir tahun j (dengan asumsi bahwa tahun 0 adalah sekarang).

$$
\begin{array}{cc}
 & j \\
 & \begin{array}{ccc} 1 & 2 & 3 \end{array} \\
i \; \begin{array}{c} 0 \\ 1 \\ 2 \end{array} & \begin{bmatrix} 12 & 27 & 47 \\ & 15 & 32 \\ & & 18 \end{bmatrix}
\end{array}
$$

Masalahnya adalah menentukan pada jam berapa (jika ada) traktor harus diganti untuk meminimalkan total biaya. Rumuskan ini sebagai masalah jarak terpendek dan selesaikan. (Masalah ini dari Hillier dan Lieberman, 1986.)

9.8 Sebuah bengkel menjual sepeda motor bekas masing-masing seharga Rp500.000. Penjualannya hanya di awal tahun ajaran dan harga belinya sama setiap tahun. Seorang siswa dapat membeli kendaraan dan menggunakannya selama empat tahun atau menggantinya setelah menggunakannya selama satu tahun atau dua tahun atau tiga tahun. Nilai tukar tambah kendaraan adalah Rp 1.500.000 setelah satu tahun, Rp 750.000 setelah dua tahun, Rp 450.000 setelah tiga tahun, dan Rp 0 setelah empat tahun. Biaya perawatan untuk kendaraan bekas adalah Rp 3.000.000, Rp 6.000.000, Rp 9.000.000, dan Rp 12.000.000, masing-masing, selama empat tahun. Masalah bagi seorang siswa yang tertarik untuk membeli sepeda motor dari dealer untuk digunakan selama empat tahun di perguruan tinggi adalah untuk membuat keputusan apakah sepeda yang dibeli sebagai mahasiswa baru harus disimpan selama empat tahun atau harus diganti sehingga total biaya adalah minimum. Jika penggantian diperlukan, pada interval berapa? Rumuskan masalah ini sebagai masalah jarak terpendek dan selesaikan.

9.9 Jika bobot sisi-sisi suatu graf terhubung semuanya merupakan bilangan positif yang berbeda, apakah benar terdapat pohon merentang minimal yang unik dalam graf tersebut? Benarkah jalur terpendek antara dua simpul adalah unik?

9.10 Suatu jaringan dikatakan memenuhi pertidaksamaan segitiga jika untuk setiap tiga busur berbeda (i, j), (j, k), dan (i, k) berat busur (i, k) tidak melebihi jumlah berat kedua busur lainnya. Lintasan berarah P dari simpul x ke simpul y adalah lintasan dengan panjang minimum jika jumlah busur di P tidak melebihi jumlah busur pada lintasan lain dari x ke y. Apakah perlu jalur terpendek dari x ke y adalah jalur panjang minimum?

LAMPIRAN
APA ITU KELENGKAPAN NP?

A.1 MASALAH DAN INSTALASINYA

Secara informal, kami selalu membedakan antara masalah dan contoh dari masalah. Misalnya, "menyelesaikan sistem persamaan linear" adalah masalah yang contoh akan "diberikan matriks A $m \times n$ dan matriks B $m \times 1$ menguji apakah ada matriks $n \times 1$ x sedemikian rupa sehingga $Ax = B$ dan jika jawabannya adalah ya dapatkan vektornya." Di sini A dan B adalah "input" dan x adalah "solusi" atau "output." Di sisi lain, contoh masalah keputusan "apakah sistem linier memiliki solusi?" akan menjadi "apakah ada x sedemikian rupa sehingga $Ax = B$?" Outputnya sekarang adalah "ya" atau "tidak." Masalah dan masalah keputusan sehingga dilihat memiliki jumlah kasus yang tak terbatas.

Pendekatan yang lebih formal untuk konsep "masalah" dan "contoh masalah" seperti yang didefinisikan dalam Schrijver (1986) adalah sebagai berikut. Alfabet adalah himpunan berhingga L yang unsur-unsurnya berupa simbol atau huruf. Urutan berhingga simbol dari L disebut string atau kata. Himpunan semua string atau kata dari L dilambangkan dengan L*. Ukuran atau panjang string adalah jumlah komponen di dalamnya. Jika L = {a, b, c}, panjang x = abbaa adalah 5 meskipun x hanya terdiri dari dua simbol. String ukuran 0 disebut string kosong, dilambangkan dengan --.

Ada beberapa cara penyandian bilangan rasional, vektor, matriks, sistem pertidaksamaan, sistem linier, graf (representasi matriks), dan sebagainya, sebagai untaian simbol dari abjad tetap seperti L = {0, 1}. Dalam diskusi kita sekarang, kita akan melewatkan rincian pengkodean tersebut. Untuk lebih jelasnya, lihat Garey dan Johnson (1979).

Soal p adalah himpunan bagian dari L* × L*. Untuk masalah p apa pun, kita memiliki masalah metamatematika yang sesuai: Diberikan string z di L*, temukan string y di L* sedemikian rupa sehingga (z, y) ada di p atau laporkan bahwa tidak ada string y seperti itu. Di sini string z disebut instance atau input dari masalah dan y disebut solusi atau output. Masalah p disebut masalah keputusan jika setiap kali (z, y) ada di p, maka y adalah string kosong. Jika L*(p) = {z L* : (z,) p} , masalah metamatematika yang sesuai adalah: Diberikan string z di L*, apakah itu milik L*(p)?

Jadi himpunan {((A, B), x) : A adalah matriks, B dan x adalah vektor kolom sehingga Ax = B} adalah himpunan bagian dari L* × L* (di mana L = {0, 1}) , mendefinisikan masalah p yang (A, B) adalah turunannya dan x adalah solusinya. Masalah ini dapat ditulis dalam bahasa metamatematika sebagai berikut: Diberikan string (A, B), cari string x (jika ada) sedemikian rupa sehingga $Ax = B$. Demikian pula, himpunan {((A, B),) : A adalah matriks, fi adalah vektor kolom sedemikian rupa sehingga $Ax = B$ untuk setidaknya satu vektor kolom x} adalah masalah keputusan versi metamatematikanya adalah sebagai berikut: Diberikan matriks A dan vektor kolom b, apakah ada kolom vektor x sehingga $Ax = B$?

A.2 UKURAN INSTANSI

Jika bilangan rasional q berbentuk m/n (di mana m dan n relatif prima, m adalah bilangan bulat, dan n adalah bilangan bulat positif), ukuran q didefinisikan sebagai ukuran (?) = 1 + plafon dari [log(m + 1)] + plafon [log(n + 1)]. Ada cara lain untuk mendefinisikan ukuran bilangan rasional. Tetapi dapat ditunjukkan bahwa sebagian besar definisi ini adalah "ekuivalen linier" (lihat Bagian A.5 untuk definisi kesetaraan linier). Jadi ukuran bilangan bulat positif sebanding dengan logaritmanya dan bukan nilainya. Lihat Garey dan Johnson (1979) lagi untuk ukuran instans dan encoding. Jika A adalah vektor dengan n komponen rasional, kita mendefinisikan ukuran (A) = n + jumlah ukuran semua komponen A. Demikian pula, jika M adalah matriks rasional m × n, tentukan ukuran (M) = mn + jumlah ukuran semua elemen matriks. Ukuran persamaan linier ax = b atau pertidaksamaan linier ax b adalah 1 + ukuran(a) + ukuran(b). Ukuran sistem linier Ax = B adalah 1 + ukuran(A) + ukuran(B). Ukuran grafik adalah ukuran matriks kejadiannya.

A.3 ALGORITMA UNTUK MEMECAHKAN MASALAH

Algoritma untuk memecahkan masalah, dalam pengertian informal, adalah urutan instruksi yang terbatas untuk mendapatkan output untuk input masalah yang diberikan. Ini adalah prosedur langkah demi langkah untuk memecahkan masalah. Dengan demikian diberikan contoh z di L*, algoritma untuk masalah p menentukan output y di L* sedemikian rupa sehingga (z, y) ada di p atau berakhir tanpa memberikan output jika tidak ada string y seperti itu. Dimungkinkan untuk mendefinisikan suatu algoritma dalam arti yang lebih formal dalam hal mesin Turing atau program komputer dalam beberapa bahasa pemrograman. Untuk tujuan kami, konsep informal algoritma ini sudah cukup. Kami menyebutkan secara sepintas bahwa ada masalah yang terdefinisi dengan baik dalam matematika yang tidak ada algoritmanya. Masalah tidak dapat diputuskan jika tidak ada algoritma yang akan menyelesaikan setiap contoh masalah. Terbukti pada tahun 1970 oleh matematikawan Rusia berusia 22 tahun Yuri Matiyasevich bahwa masalah keputusan yang dikenal sebagai masalah kesepuluh Hilbert, yang menanyakan apakah persamaan polinomial di lebih dari satu variabel dengan koefisien bilangan bulat memiliki solusi bilangan bulat, adalah masalah yang tidak dapat diputuskan.

Masalah yang tidak dapat diputuskan yang paling terkenal dalam ilmu komputer adalah masalah penghentian: Diberikan program komputer dengan inputnya, akankah program itu berhenti? Ketika kita mengatakan bahwa masalah penghentian tidak dapat diputuskan, yang kita maksudkan adalah bahwa tidak ada algoritma yang akan memutuskan apakah program komputer arbitrer akan masuk ke loop tak terbatas saat mengerjakan input yang diberikan. Sebuah referensi yang sangat baik untuk topik unde-cidability dan item terkait yang menarik adalah buku oleh Lewis dan Papadimitriou (1981).

A.4 KOMPLEKSITAS ALGORITMA

Jika kita memiliki dua algoritme untuk menyelesaikan setiap contoh masalah, wajar untuk membandingkannya untuk mengetahui apakah yang satu lebih baik atau lebih efisien daripada yang lain. Untuk tujuan ini kita harus mengukur jumlah pekerjaan yang dilakukan oleh algoritma, yang merupakan jumlah "operasi dasar" yang diperlukan untuk menyelesaikan masalah menggunakan algoritma. Berikut adalah beberapa contoh operasi dasar. Operasi dasar dalam masalah pengurutan adalah perbandingan dua angka dalam daftar entri yang diberikan dan dengan demikian pekerjaan yang dilakukan dalam masalah pengurutan adalah jumlah perbandingan. Dalam masalah yang melibatkan perkalian dan penjumlahan, kita dapat mengambil perkalian sebagai operasi dasar karena perkalian lebih sulit daripada penjumlahan. Jadi pekerjaan yang dilakukan dalam mengalikan dua matriks n × n paling banyak n3 (Lihat Bagian 3.5.)

Jika A adalah algoritma untuk menyelesaikan (setiap contoh) masalah dan jika x adalah turunan dari masalah, jumlah operasi dasar yang diperlukan untuk menyelesaikan x menggunakan A dilambangkan dengan wA(x) . Karena kita tertarik pada efisiensi kerja sebenarnya dari algoritme, penting bahwa ukuran yang kita pilih untuk menentukan pekerjaan yang dilakukan tidak tergantung pada komputer yang digunakan, program komputer tertentu, bahasa pemrograman, dan detail implementasi lainnya. Biasanya, pekerjaan yang dilakukan diambil sebagai fungsi dari ukuran contoh masalah. Sekarang jika dua contoh memiliki ukuran yang sama, itu tidak berarti bahwa pekerjaan yang dilakukan adalah sama untuk keduanya. Jadi kita harus menggabungkan dalam satu atau lain cara pekerjaan yang dilakukan untuk semua contoh dengan panjang yang sama.

Salah satu cara untuk melakukan ini adalah dengan mengambil pendekatan kasus terburuk. Jadi, kompleksitas kasus terburuk dari algoritma A untuk menyelesaikan masalah p didefinisikan sebagai fA(n), di mana fA(n) = Max {wA(x) : x adalah turunan dari p dan ukuran x adalah n} Kita mungkin mengambil pendekatan yang berbeda sebagai berikut. Misalkan h(x) adalah probabilitas bahwa sebuah instance x dari masalah diambil sebagai kandidat untuk input. Maka kompleksitas kasus rata-rata adalah jumlah semua suku h(x) . wA(x) di mana x adalah turunan dari ukuran n. Dalam apa mengikuti kompleksitas berarti kompleksitas dalam kasus terburuk.

A.5 NOTASI "BIG OH" ATAU O(.)

Misalkan f dan g adalah dua fungsi dari himpunan bilangan asli ke himpunan bilangan real nonnegatif. Jika ada konstanta positif c dan bilangan asli n0 sehingga f(n) c • g(n) untuk semua n n0, kita tulis “ f is O(g)” atau “f = O(g)” atau “f(n) adalah O(g(n))” atau “f(n) = O(g(n))” dan katakan (seperti dalam Wilf, 1986) bahwa “f(n) besar oh dari g (n)." Dua fungsi f dan g ekivalen linear jika f = O(g) dan g = O(f). Kita tulis f < g jika f dan g ekivalen linear. Relasi >< yang didefinisikan adalah relasi ekivalensi dan laju pertumbuhan f adalah kelas ekivalennya, yang dapat diwakili oleh anggota kanonik dari kelas tersebut. Sebagai contoh, misalkan f(n) = 5n2 + 9n + 7 dan g(n) = 8n2 + 23. Maka mudah untuk menunjukkan bahwa f >< g dan perwakilan tipikal dari kelas ekivalensi yang dimiliki f dan g adalah fungsi p(n) = n2. Jadi kita tulis f(n) = O(n2) dan g(n) = O(n2). Sekarang perhatikan fungsi h(n) = 4n3 + 9n. Maka f(n) adalah O(h(n)) tetapi h(n) jelas bukan dari O(f(n)). Perhatikan bahwa notasi oh besar hanya memberikan batas atas. Jika f(n) adalah O(nk), sangat mungkin bahwa f(n) adalah O(nr) untuk beberapa r kurang dari k. Pada saat yang sama f(n) adalah O(nr) untuk semua r k. Misal c = lim [f(n)/g(n)] sebagai n menuju plus tak terhingga.

Maka dapat dengan mudah diverifikasi bahwa:

1. Jika c berhingga dan bukan nol, f dan g ekivalen linear.

2. Jika c adalah nol, f adalah O(g) tetapi g bukanlah O(f) .
3. Jika c tak hingga, g adalah O(f) tetapi f bukan O(g) .

Sebagai akibatnya, kita katakan bahwa f orde lebih rendah dari g (atau ekuivalen, g orde lebih tinggi dari f) jika c = 0. Dengan menggunakan uji rasio ini, kita dapat menetapkan bahwa jika k bilangan bulat positif, nk lebih tinggi urutan dari log n, 2n adalah urutan yang lebih tinggi dari nk dan n! adalah urutan yang lebih tinggi dari 2n.

A.6 MASALAH MUDAH DAN MASALAH SULIT

Suatu algoritme A untuk memecahkan masalah p disebut algoritme polinomial atau algoritme waktu polinomial jika kompleksitas kasus terburuknya fA(n) adalah O(nk) untuk beberapa bilangan bulat positif tetap k. Jadi algoritma dengan kompleksitas n log n adalah algoritma polinomial karena n log n adalah O(n2) . Algoritma yang kompleksitasnya melanggar semua batas polinomial disebut sebagai algoritma eksponensial. Suatu algoritma dengan kompleksitas f(n) adalah eksponensial jika dan hanya jika ada bilangan positif a dan b, bilangan p dan q lebih besar dari 1, dan bilangan bulat positif n0 sehingga a . pn f(n) b . qn untuk semua n n0 .

Beberapa contoh laju pertumbuhan algoritma eksponensial adalah kn (k > 1), n!, nn dan nlogn. Untuk membahas apakah suatu masalah mudah atau tidak, pertama-tama harus diputuskan di mana harus menarik garis antara masalah yang mudah dan sulit. Perbedaan antara fungsi eksponensial dan polinomial menjadi jelas jika kita mengambil sudut pandang asimtotik: Polinomial tumbuh lebih lambat daripada fungsi eksponensial. Jadi algoritma polinomial dengan laju pertumbuhan nk (bahkan ketika k besar) efisien dibandingkan dengan fungsi eksponensial. Dengan kata lain, untuk masalah yang cukup besar, algoritma polinomial yang dieksekusi pada komputer paling lambat akan menyelesaikan masalah lebih cepat daripada algoritma eksponensial pada komputer tercepat. Selanjutnya, dalam beberapa kasus, suatu algoritma untuk memecahkan suatu masalah dapat diperoleh dengan menggabungkan beberapa algoritma untuk submasalah yang lebih sederhana. Jika masing-masing algoritme submasalah ini polinomial, maka algoritme masalah utama juga polinomial karena kelas dari semua polinomial tertutup dalam penjumlahan, perkalian, dan komposisi fungsi.

Jadi konsensus di antara para ilmuwan komputer adalah untuk mengatakan bahwa masalah itu mudah jika ada algoritma polinomial yang akan menyelesaikan setiap contoh masalah. Ide ini awalnya karena Edmonds (1965b). Perhatikan bahwa tidak masuk akal untuk mengatakan bahwa suatu algoritma baik jika kompleksitasnya O(nk) ketika k besar. Dalam hubungan ini, komentar berikut dari Papadimitriou dan Steiglitz (1982) layak untuk direproduksi: “Tes bahwa algoritma waktu polinomial adalah 'baik' tampaknya melemah ketika didorong ke ekstrem. Pengalaman, bagaimanapun, datang untuk mendukungnya. Untuk sebagian besar masalah, setelah algoritma polinomial-waktu ditemukan, derajat polinomial dengan cepat mengalami serangkaian penurunan seiring berbagai peneliti memperbaiki gagasan tersebut. Biasanya laju pertumbuhan akhir adalah O(n3) atau lebih baik.”

Untuk menghargai perbedaan dalam laju pertumbuhan algoritme polinomial dan laju pertumbuhan tirani algoritme eksponensial, pertimbangkan skenario berikut: Misalkan operasi dasar (setiap langkah) di komputer memerlukan sepersejuta detik waktu komputer . Jika n = 50, waktu komputasi untuk algoritma dengan kompleksitas n2, n3, 2n, dan 3n masing-masing adalah 0,0025 detik, 0,125 detik, 35,7 tahun, dan (2)(109) abad. Jika n = 100, waktu ini berturut-turut adalah 0,01 detik, 1 detik, 248 abad, dan 370 abad.

Telah disebutkan sebelumnya bahwa suatu masalah (dapat dibuktikan) tidak dapat diputuskan jika dapat dibuktikan bahwa tidak ada algoritma yang akan menyelesaikan setiap contoh masalah. Alih-alih menanyakan apakah suatu masalah terbukti tidak dapat diputuskan atau tidak, pertanyaan dasar dalam teori kompleksitas komputasi menanyakan seberapa sulit atau sulit untuk memecahkan masalah terbukti sulit jika dapat dibuktikan bahwa algoritma apa pun yang akan menyelesaikan (setiap contoh) masalah adalah algoritma eksponensial. Masalah seperti itu memang ada, tetapi mereka agak kabur. Lihat Lewis dan Papadimitriou (1981) untuk lebih jelasnya. Masalah yang tidak diketahui algoritme polinomialnya dan yang diduga tidak ada algoritmenya disebut masalah yang tidak dapat diselesaikan.

Masalah menemukan jalur terpendek antara titik dan titik lain dalam graf terhubung adalah masalah yang mudah (algoritma Dijkstra adalah polinomial), tetapi masalah menemukan jalur sederhana terpanjang antara dua titik tidak dapat diselesaikan karena tidak ada yang tahu algoritma untuk menyelesaikan ini yang secara substansial lebih cepat daripada menghitung semua jalur yang mungkin antara keduanya dan memilih yang optimal.

Jadi jika ada algoritma polinomial untuk menyelesaikan masalah p, kita dapat mengatakan bahwa “mudah untuk menyelesaikan p.” Tetapi bagaimana seseorang dapat menunjukkan bahwa “sulit (tidak mudah) untuk memecahkan suatu masalah”? Seseorang dapat membuktikan bahwa masalah p sama sulitnya dengan masalah q. Jadi setiap kali tidak ada algoritma polinomial yang diketahui untuk memecahkan masalah, kita harus memeriksa masalah itu dalam semangat ini. Dengan kata lain, untuk membuktikan bahwa masalah yang diberikan sulit, tidak cukup untuk menyatakan bahwa sejauh ini tidak ada algoritma polinomial yang ditemukan untuk menyelesaikannya.

Dibutuhkan beberapa teknik matematika yang canggih untuk menunjukkan bahwa kompleksitas algoritma apa pun yang mungkin untuk masalah tersebut tidak dapat dibatasi di atas oleh polinomial. Teknik seperti itu sekarang ditemukan oleh para ilmuwan komputer dengan munculnya perkembangan terbaru dalam teori kompleksitas komputasi. Pengenalan konsep kelengkapan NP merupakan tonggak penting dalam bidang ini.

A.7 KELAS P DAN KELAS NP

Selanjutnya kita akan mengasumsikan bahwa masalah yang kita pertimbangkan adalah semua masalah keputusan. Ini adalah masalah yang hasilnya adalah "ya" atau "tidak". Contoh masalah keputusan: (1) Apakah ada siklus Hamilton dalam graf terhubung yang diberikan? dan (2) Apakah ada jalur sederhana dari simpul v ke simpul w lain dalam jaringan yang terhubung sedemikian rupa sehingga total panjang jalur ini tidak melebihi jumlah tertentu?

Kami tertarik untuk mengklasifikasikan masalah keputusan menurut kompleksitasnya. Masalah keputusan termasuk dalam kelas P jika ada algoritma polinomial untuk menyelesaikan setiap contoh masalah. Jika suatu masalah memiliki algoritma polinomial, maka jelas masalah keputusan yang sesuai juga memiliki algoritma polinomial.

Perhatikan bahwa kita dapat menyatakan bahwa masalah arbitrer ada di kelas P hanya setelah kita memiliki bukti bahwa ada algoritma polinomial untuk menyelesaikannya. Fakta bahwa masalah pemrograman linier adalah anggota dari kelas P didirikan hanya satu dekade yang lalu, ketika Khachiyan (1979) keluar dengan algoritma ellipsoid-nya. Selanjutnya diperoleh algoritma yang lebih efisien oleh Karmarkar (1984) dengan menggunakan metode titik interior. Untuk deskripsi yang jelas tentang perkembangan ini, lihat Schrijver (1986).

Sekarang pertimbangkan masalah keputusan yang statusnya adalah sebagai berikut: (1) setidaknya ada satu algoritme eksponensial untuk menyelesaikannya, dan (2) sejauh ini belum ada yang membuktikan bahwa setiap algoritme yang akan dipecahkan selalu eksponensial. Dengan kata lain, ini adalah masalah keputusan yang dapat diputuskan dan kita tidak tahu apakah itu terbukti sulit atau tidak. Apa yang kita lakukan dengan masalah dalam kategori ini?

Pada tahap inilah kami memperkenalkan kelas masalah keputusan yang berisi kelas P. Masalah keputusan milik kelas NP jika ada algoritma polinomial untuk memverifikasi output "ya" dari masalah itu. Akronim NP adalah untuk "polinomial nondeterministik." Untuk detail tentang algoritma nondeterministik, lihat Garey dan Johnson (1979). Algoritma nondeterministik sama sekali bukan algoritma probabilistik atau acak.

Jelas bahwa setiap masalah keputusan dengan algoritma polinomial ada di kelas NP. Mengenai masalah tipikal p di NP yang sejauh ini belum ada yang memperoleh algoritma polinomial, ada tiga alternatif yang saling eksklusif: (1) algoritma polinomial untuk p akan ditemukan, (2) akan dibuktikan bahwa p tidak akan memiliki algoritma polinomial, dan (3) status p tidak akan pernah diselesaikan.

Sejauh ini belum ada yang membuktikan bahwa ada masalah di NP yang tidak ada di P. Tidak diketahui apakah P = NP atau apakah P benar terkandung dalam NP. Ini adalah situasi yang membuat frustrasi karena banyak masalah praktis milik kelas NP. Jika P = NP, masuk akal untuk mencoba mendapatkan algoritme polinomial untuk masalah dalam NP yang sejauh ini belum ditemukan algoritme efisien. Di sisi lain, jika kita dapat menetapkan bahwa masalah tertentu di NP tidak ada di P, kita tidak perlu repot mencari algoritma yang efisien untuk menyelesaikannya. Tetapi dengan tidak adanya bukti, kami tidak dapat mengabaikan upaya kami untuk mendapatkan algoritma polinomial untuk menyelesaikan masalah ini karena selalu ada kemungkinan kecil bahwa di suatu tempat di luar sana ada algoritma polinomial untuk menyelesaikan masalah yang menunggu untuk ditemukan!

Berikut adalah contoh masalah di NP. Pertimbangkan masalah keputusan yang contohnya adalah sebagai berikut: "Apakah bilangan bulat positif n bilangan komposit?'' Ketika n besar, kita tidak dapat dengan mudah menjawab pertanyaan ini. Namun, jika kita dapat menunjukkan dua bilangan bulat positif p dan q (keduanya lebih besar dari 1) sehingga n = pq, maka siapa pun dapat dengan mudah mengatakan jawabannya adalah “ya” karena perkalian dua bilangan dapat dilakukan dalam waktu polinomial. Di sisi lain, sama sekali tidak jelas apakah masalah keputusan “apakah bilangan bulat positif n bilangan prima?” termasuk dalam kelas NP. Hal ini dibuktikan oleh Pratt (1975) bahwa memang demikian adanya.

Sesuai dengan setiap masalah keputusan p, selalu ada masalah keputusan pelengkap p′. Masing-masing saling melengkapi. Soal “apakah benar n prima?” dan “benarkah n bukan prima?” saling melengkapi. Dalam graf terhubung G masalah "benarkah ada siklus Hamilton di G?" dan “benarkah tidak ada siklus Hamilton di G?” saling melengkapi. Perhatikan bahwa masalah sebelumnya dalam teori graf ada di NP karena mudah untuk memverifikasi jawaban "ya". Tetapi masalah terakhir tidak ada di NP karena untuk memverifikasi jawaban "ya" dalam kasus ini, kita harus menghitung semua siklus yang mungkin. Kebetulan, kami memiliki masalah keputusan yang tidak ada di NP. Jadi jika masalah ada di NP tidak perlu masalah pelengkapnya ada di NP. Masalah keputusan termasuk dalam kelas Co-NP jika masalah komplementernya ada di kelas NP. Masalah keputusan dikatakan terkarakterisasi dengan baik jika berada pada NP dan Co-NP. Jelas, setiap masalah di P ditandai dengan baik. Tidak diketahui apakah setiap masalah yang berkarakteristik baik ada di P. Soal-soal “adalah n prima?” dan "apakah n komposit?" keduanya dikarakterisasi dengan baik karena teorema Pratt.

Juga tidak diketahui apakah NP = Co-NP. Jika NP terkandung dalam Co-NP (atau jika Co-NP terkandung dalam NP), maka NP = Co-NP = NP Co-NP. Juga, jika P = NP, semua himpunan bertepatan.

A.8 TRANSFORMASI POLINOMIAL DAN KELEPATAN NP

Masalah mudah di NP ada di P. Mereka berada di satu sisi spektrum. Karena tidak diketahui apakah P = NP atau tidak, wajar untuk bertanya apakah kita dapat mengumpulkan semua masalah "sulit" NP dalam satu kelas dan menempatkan kelas ini di sisi lain spektrum. Pertama-tama, apa yang seharusnya menjadi nama kelas yang terhormat ini? Tampaknya Donald Knuth dari Universitas Stanford mensurvei rekan-rekannya pada tahun 1974 untuk menemukan nama yang tepat. Ada banyak saran: tangguh, Hercules, sulit, luar biasa, keras kepala, dan sebagainya. Akhirnya, sebuah konsensus: Sebut saja kelas masalah NP-complete — dan nama ini melekat di antara ilmuwan komputer, ahli logika, dan ahli matematika.

Ide dasar dalam teori kelengkapan NP adalah transformasi polinomial. Masalah keputusan p dapat ditransformasikan secara polinomial ke masalah keputusan q jika dua kondisi berikut terpenuhi: (1) terdapat fungsi f(x) yang akan mentransformasikan setiap instance x dari p menjadi instance f(x) dari q sedemikian rupa sehingga jawaban untuk x adalah “ya” jika dan hanya jika jawaban untuk f(x) adalah “ya” dan (2) terdapat algoritma yang efisien untuk menghitung f(x) untuk setiap x.

Berikut adalah contoh masalah yang dapat ditransformasikan secara polinomial menjadi masalah lain. Ingatlah bahwa sebuah klik dalam graf G adalah subgraf lengkap dari G. Banyaknya simpul dalam sebuah klik adalah ukurannya. Masalah klik p dinyatakan sebagai berikut: Apakah ada klik dengan ukuran yang ditentukan dalam grafik? Untuk himpunan berhingga yang diberikan, kumpulan himpunan bagian dikatakan menutupi himpunan jika setiap elemen dalam himpunan tersebut dimiliki oleh setidaknya satu himpunan dalam kumpulan tersebut. Himpunan yang mencakup masalah q dinyatakan sebagai berikut: Diberikan suatu himpunan berhingga X, himpunan C dari himpunan bagian dari X, dan bilangan bulat positif m, apakah ada himpunan bagian C′ yang terdiri dari m himpunan tersebut sehingga C′ mencakup X?

Masalah p dapat diubah menjadi q sebagai berikut. Misalkan G = (V, E) adalah graf terhubung dengan m sisi di mana V = {1, 2, 3, . . . , n}. Maka komplemennya G′ = (V, E′) memiliki r rusuk, di mana r = [n(n – 1)/2]. Misalkan E′ = {e1 e2, . . . , er} adalah himpunan yang akan dicakup. Misalkan Si adalah himpunan sisi di G′ yang datang di titik i. Kita ambil C = {Si : i = 1, 2, . . . , n} sebagai kumpulan himpunan bagian dari E′ yang tersedia. Sangat mudah untuk melihat bahwa jika G memiliki klik berukuran k, maka subkumpulan C′ dari (n – k) himpunan bagian dapat dipilih dari C yang akan menutupi himpunan E′. Khususnya, jika W = {1, 2, . . . , k} adalah himpunan simpul yang membentuk klik di G, himpunan C′ = {Sk +1 Sk +2 , . . . , Sn} akan menjadi penutup untuk E′.

Definisi transformabilitas polinomial menunjukkan ketidaksetaraan berikut: Jika p dapat ditransformasikan secara polinomial ke q, (kompleksitas p) (kompleksitas f) + (kompleksitas q) . Selanjutnya, jika kompleksitas f tidak signifikan dibandingkan dengan kompleksitas p dan q, kita dapat menulis, dalam pengertian asimtotik, (kompleksitas p) (kompleksitas q) .

Masalah keputusan p adalah NP-hard jika setiap masalah dalam NP dapat ditransformasikan ke dalam polinomial. Dengan kata lain, masalah NP-hard tidak bisa lebih mudah daripada masalah apa pun di NP. Masalah dalam NP yang merupakan NP-hard dikatakan sebagai NP-complete. Kelas masalah NP-lengkap dengan demikian merupakan perpotongan dari kelas NP dan dan kelas NP-keras. Jadi, jika ada algoritma yang efisien untuk menyelesaikan setiap contoh masalah NP-complete tertentu, setiap masalah di NP memiliki algoritma polinomial. Kelas masalah NP-complete dilambangkan dengan NPC. Untuk menunjukkan bahwa masalah p ada di NPC harus dibuktikan bahwa (1) p ada di NP dan (2) setiap masalah di NP dapat ditransformasikan secara polinomial menjadi p. Perhatikan bahwa kompleksitas masalah dalam NPC terkait erat dengan dugaan bahwa P adalah himpunan bagian yang tepat dari NP.

Fakta bahwa kelas NPC tidak kosong ditetapkan oleh Cook (1971) dalam makalahnya dengan menunjukkan masalah di NP sedemikian rupa sehingga setiap masalah di NP dapat ditransformasikan ke dalam polinomial. Masalah ini dikenal sebagai masalah satisfiability. Masalah ini berasal dari logika matematika dan aplikasi dalam teori switching. Namun, dapat dinyatakan sebagai teka-teki kombinatorial sederhana seperti dalam Karp (1986):

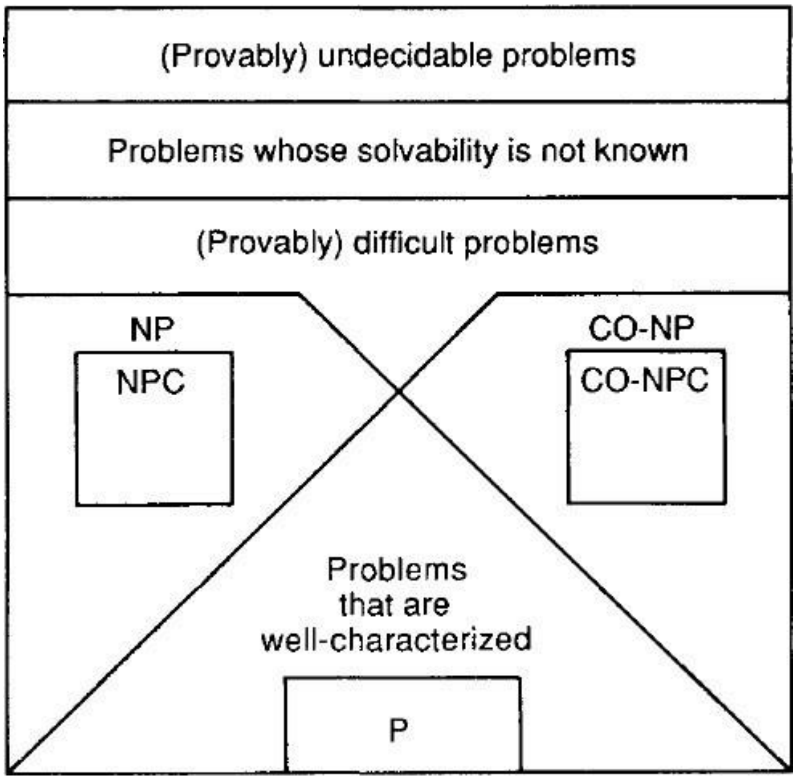
> Mengingat beberapa urutan huruf besar dan kecil, apakah mungkin untuk memilih huruf dari setiap urutan tanpa memilih versi huruf besar dan kecil dari urutan yang sama? Misalnya, jika barisannya adalah Abc, BC, aB, dan ac, maka dimungkinkan untuk memilih A dari barisan pertama, B dari barisan kedua dan ketiga, dan c dari barisan keempat; perhatikan bahwa huruf yang sama dapat dipilih lebih dari satu kali, asalkan kami tidak memilih versi huruf besar dan kecilnya. Contoh di mana tidak ada cara untuk membuat pilihan yang diperlukan diberikan oleh empat urutan AB, Ab, aB dan ab.
>
> Masalah kepuasan jelas ada di NP, karena mudah untuk memeriksa apakah pemilihan huruf yang diusulkan memenuhi kondisi masalah. Cook membuktikan bahwa, jika masalah satisfiability dapat diselesaikan dalam waktu polinomial, maka setiap masalah dalam NP dapat diselesaikan dalam waktu polinomial, sehingga P = NP. Jadi kita melihat bahwa masalah yang tampaknya aneh dan tidak penting ini adalah masalah kombinatorial pola dasar; itu memegang kunci untuk solusi efisien dari semua masalah di NP.

“Pintu air” terbuka setelah terbukti bahwa NPC kelas tidak kosong. Dengan membangun serangkaian transformasi polinomial Karp (1972a) menghasilkan daftar 20 atau lebih masalah di NPC. Ditunjukkan dalam makalah ini bahwa sebagian besar masalah kombinatorial klasik seperti pengepakan, penutup, pencocokan, partisi, perutean, dan sebagainya, ada di NPC. Daftarnya meliputi: (1) Apakah graf tertentu Hamiltonian?; (2) Apakah mungkin mewarnai simpul-simpul dari suatu graf dengan k warna sehingga tidak ada dua simpul bertetangga yang memiliki warna yang sama?; (3) Diberikan himpunan bilangan {ni : i = 1, 2, . . ., k} dan suatu bilangan s, apakah suatu himpunan bagian dari bilangan tersebut dijumlahkan dengan tepat s?; dan (4) Apakah ada klik dengan ukuran tertentu dalam grafik?

Jika p adalah masalah di NP, untuk menunjukkan bahwa p ada di NPC, cukup dibuktikan bahwa beberapa masalah yang diketahui di NPC dapat ditransformasikan secara polinomial. Ada ribuan masalah yang sekarang diketahui sebagai NP-lengkap dan "suku" mereka terus meningkat, seperti yang dapat dilihat dari publikasi hasil baru di bidang ini dalam beberapa tahun terakhir. Untuk penjelasan yang menarik tentang dunia kelengkapan NP, seseorang harus merujuk pada buku karya Garey dan Johnson (1979) dan kolom Johnson tentang topik ini yang berjudul “Kelengkapan NP: Panduan yang Berkelanjutan,” yang muncul secara teratur di Journal of Algorithms . Sekarang rutin untuk menyelidiki apakah masalah yang tampaknya sulit adalah NP-complete.

Ingatlah bahwa masalah di NP dicirikan dengan baik jika komplemennya juga ada di NP. Jika dapat dibuktikan adanya masalah pada NPC yang terkarakterisasi dengan baik, maka dapat ditunjukkan bahwa NP = Co-NP. Upaya untuk menunjukkan bahwa pelengkap dari beberapa masalah lengkap NP standar ada di NP tidak membuahkan hasil. Juga, tidak ada bukti untuk percaya bahwa kedua kelas NP dan Co-NP ini bertepatan. Oleh karena itu dugaan: NPC kelas dan kelas W dari masalah yang dicirikan dengan baik adalah terputus-putus. (Sudah diketahui untuk beberapa waktu bahwa masalah pemrograman linier LP adalah masalah yang dikarakterisasi dengan baik. Jadi diduga bahwa LP tidak ada di NPC bahkan sebelum ditemukannya algoritma ellipsoid, yang membuktikan bahwa tidak hanya LP yang tidak ada di NPC tetapi dalam P.) Demikian pula, jika Co-NPC adalah kelas pelengkap dari semua masalah NP-lengkap, kelas Co-NPC dan kelas W juga terputus. Dengan demikian topografi dugaan kelas masalah keputusan adalah seperti yang digambarkan pada Gambar A.8.1.

Akhirnya, bahkan jika kita berasumsi bahwa setiap masalah NP-complete terbukti sulit, tetap ada kelas masalah yang statusnya belum terselesaikan. Misalnya, pertimbangkan masalah keputusan p di NP sedemikian rupa sehingga (1) sejauh ini tidak ada algoritma polinomial yang diperoleh untuk menyelesaikan setiap instance p, dan (2) sejauh ini tidak ada bukti bahwa p ada di NPC. Secara khusus, status masalah yang dicirikan dengan baik (yang tidak mungkin dalam NPC) yang sejauh ini tidak ada algoritma polinomial yang diperoleh tetap tidak terselesaikan. Anggota khas dari kelas ini: "Apakah bilangan bulat positif yang diberikan merupakan bilangan prima?"



A.9 MENGATASI MASALAH KERAS

Masalah yang dipertimbangkan sejauh ini adalah masalah keputusan. Dalam masalah optimasi kombinatorial mungkin ada banyak solusi (solusi yang layak) dan setiap solusi akan memiliki bilangan real yang terkait dengannya yang disebut nilai solusi. Tujuan dari masalah adalah untuk mendapatkan solusi (solusi layak optimal) yang nilainya optimal. Sesuai dengan masalah optimasi seperti itu, ada masalah keputusan untuk menentukan apakah masalah optimasi memiliki solusi dengan nilai yang lebih baik daripada bilangan real yang diberikan. Jelas, masalah keputusan yang terkait dengan masalah optimasi kombinatorial tidak bisa lebih sulit daripada masalah optimasi itu sendiri. Jadi jika masalah “Apakah graf Hamiltonian?” sulit, maka masalah "menemukan siklus Hamiltonian yang optimal dalam grafik" juga sulit. Lebih tepatnya, ini berarti bahwa jika masalah keputusan yang terkait dengan masalah optimasi adalah NP-complete, masalah optimasi adalah NP-hard.

Sekarang proposisi yang membuktikan bahwa masalah keputusan (yang sesuai dengan masalah optimasi) ada di NPC menghilangkan untuk semua tujuan praktis kemungkinan memperoleh algoritma yang efisien untuk menyelesaikan setiap contoh masalah optimasi. Tetapi faktanya tetap bahwa banyak masalah optimasi kombinatorial yang muncul di beberapa bidang dalam sains, teknik, dan riset operasi adalah NP-hard. Jadi wajar untuk bertanya: Bagaimana kita mengatasi masalah yang begitu sulit? Secara garis besar, ada dua pendekatan. Salah satunya adalah pendekatan heuristik: Kemungkinan masalahnya sulit karena sebagian kecil contoh sulit. Apakah mungkin untuk mendapatkan algoritma yang efisien yang akan memecahkan sejumlah besar contoh masalah? Sebuah algoritma heuristik selalu memberikan solusi yang optimal, tetapi tidak perlu efisien dalam setiap contoh. Metode simpleks untuk menyelesaikan masalah pemrograman linier adalah algoritma dari kategori ini. Pendekatan lainnya adalah untuk mengetahui apakah terdapat algoritma aproksimasi yang efisien untuk mendapatkan solusi yang layak dengan nilai yang sangat mendekati nilai optimal. Ada algoritma aproksimasi yang efisien untuk beberapa masalah NP-hard yang terkenal dalam optimasi kombinatorial. Sekali lagi, lihat Garey dan Johnson (1979) untuk lebih jelasnya.

JAWABAN UNTUK LATIHAN YANG DIPILIH

BAB 1

1.1. (a) A B = {2, 3, 5, 6, 7, 9} (b) B C = {2, 6} (c) B – A = {2, 6} (d) A – B = {9} (e) C′ = {3, 5, 7, 9} (f) X′ himpunan kosong (g) Komplemen himpunan kosong X

1.3. {a, b, c, c} = {a, b, a, b, c}

1,5. (a) A × A = {(3, 3), (4, 4), (3, 4), (4, 3)} (b) A × B = {(3, p), (3, q), (3, r), (4, p), (4, q), (4, r)} (c) B × A = {(p, 3), (p, 4), (q, 3), (q, 4), (r, 3), (r, 4)} (d) B × B = {(p, p), (q, q), (r. r), (p, q), (q, p), (p, r), (r, p), (q, r), (r, P)}

1.7. (a) A (B × A) = {3, 4, (p, 3), (p, 4), (q, 3),(q, 4), (r, 3), (r, 4)} (b) (A × A) (B × B) = {(3, 3), (4, 4), (3, 4), (4, 3), (p, 3), (p, 4), (q, 3), (q, 4), (r, 3), (r, 4)}

1.9. (a) {{a}} (b) {{a}, {b}} (c) {{a}, {b}, {c}}, {{a, b), {c}}, { {a, c}, {b}}, dan {{b, c}, {a}}

1.13. (a) Ya (b) Ya

1,19. Satu

1,23. Paling banyak delapan

1,25. (A – B) dan (B – A) keduanya kosong. Ini berarti bahwa A adalah himpunan bagian dari B dan pada saat yang sama B adalah himpunan bagian dari A. Jadi, A

= B

1.27. Ya. Misalkan x merupakan elemen sembarang dari B. Kasus 1: Misalkan x tidak ada di A. Maka x ada dalam beda simetris A dan

B. Jadi x ada dalam beda simetris A dan C. Jadi x ada di A C tetapi tidak di A C. Jadi x ada di C. Kasus 2: Misalkan x ada di

A. Maka x tidak terletak pada selisih simetris A dan B dan oleh karena itu x tidak terletak pada selisih simetris A dan C. Jadi x ada di A C yang menyiratkan x ada di C. Jadi, bagaimanapun, B adalah himpunan bagian dari A. Demikian juga C adalah himpunan bagian dari B.

1.29. (a) Domain dan kodomain keduanya adalah R dan range adalah himpunan semua bilangan real nonnegatif, (b) Tidak (c) Tidak

(d) {–2, 2} (e) Gabungan dua interval tertutup I dan J di mana I = {x: – 2 x – 1} dan J = {x : 1 x 2}

1.31. Fungsi f bukan surjeksi. Fungsi invers g(n) = (n – 5)/2 di mana n dalam f(N) adalah surjeksi.

1.33. (a) 3.3.3.3 (b) 0 (c) (3.3.3.3) – (3.2.2.2) + 3 (d) 1.1.3.3

1.35. (a) Domain = himpunan semua bilangan bulat, jangkauan = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} (b) Domain = himpunan semua bilangan bulat, jangkauan = himpunan semua bilangan bulat positif

1.37. {(p, 1), (q, 1), (r, 2)}

1.39. {(p, p), (q, q), (r, p)} bila n genap dan {(p, q), (q, p), (r, q)} bila n ganjil

1.43. Keempat konstanta memenuhi persamaan ad + b = bc + d.

1,49. (a) 2 (b) 3

1,51. Gambar lima lingkaran kecil di sisi kiri satu di bawah yang lain sehingga tidak ada dua lingkaran yang saling bersentuhan dan beri label 1, 2, 3, 4, dan 5. Kemudian gambar empat lingkaran kecil satu di bawah yang lain di sisi kanan dan beri label a, b, c, dan d. Gambarlah anak panah (1) dari 1 ke a, (2) dari 1 ke b, (3) dari 3 ke c, (4) dari 4 ke d, (5) dari 5 ke d, dan (6) dari 5 ke c .

1,53. Tidak dalam semua kasus kecuali dalam (c)

1,55. R2 = {(a, a), (a, b), (b, a), (b, b), (c, a), (c, b)} R3 = {(a, a), (a, b), (b, a), (b, b), (c, a), (c, b)}

1.62. Refleksif, tidak simetris, antisimetris, transitif

1,64. (a) Ini adalah relasi ekivalensi. Partisi yang sesuai adalah {{1, 3}, {2}, {4}}. (b) Ini bukan relasi ekivalensi, (c) Ini relasi ekivalensi dengan partisi {{1,2}, {2}, {3}, {4}}.

1,68. (a) {5k : k Z} (b) {1 + 5k : k Z} (c) {2 + 5k : k Z}

1,70. Gambaran awal dari elemen-elemen jangkauan f

1,78. Himpunan S memiliki 27 pasang seluruhnya. Ada 8 pasang di mana elemen pertama adalah himpunan kosong dan elemen kedua adalah setiap himpunan bagian dari X. Ada 12 pasangan di mana elemen pertama adalah himpunan tunggal dan 6 pasang di mana elemen pertama adalah himpunan dari dua elemen . Akhirnya, ({a, b, c}, {a, b, c}) ada di S.

1,80. {1, 2, 4, 8} dan {1, 3, 6} adalah dua rantai.

1.82. (a) 2, 3 (b) 16, 24 (c) 12, 24

1,87. Hipotesis induksi P(n) adalah pernyataan bahwa jumlah 1/1 • 2 + 1/2 • 3 + 1/3 • 4 + • • • + 1/n(n + 1) sama dengan n/(n + 1). Tujuannya adalah untuk membuktikan bahwa pernyataan ini benar untuk semua n. Jelas, P(1) benar karena 1/1 • 2 = 1/(1 + 1). Jadi langkah dasarnya terbukti. Selanjutnya kita harus membuktikan langkah induksi: jika P(k) benar untuk semua itu, maka P(k + 1) juga benar. Sangat mudah untuk melihat bahwa P(k + 1) benar karena k/(k + 1) + l/(k + 1)(k + 2) sama dengan (k + 1)/(k + 2).

1.105. f(1) = 1 dan f(n) = n + f(n-1)

1.107. Ini adalah tautologi.

1.109. Juga tidak

1.111.

p'	q'	$(p' \vee q')$	q	$(p' \vee q)' \leftrightarrow q$
T	T	T	F	F
T	F	T	T	T
F	T	T	F	F
F	F	F	T	F

1.113. (a) Memuaskan dalam semua kasus kecuali jika p, q, dan r benar dan s dan t; salah (b) Memuaskan dalam semua kasus kecuali jika p dan q benar dan r salah

BAB 2

2.1. 109

2.3. (a) 16 (b) 8 (c) 102 (d) 28

2.5. (a) 28 (b) 16 (c) 48 (d) 112

2.7. Ada 26 • 25 • 24 • 23 • 22 cara.

2.9. n(n – 1)

2.11. 26 + (26)(36) + (26)(36)2 + (26)(36)3 + (26)(36)4 + (26)(36)5

2.13. (a) 12 (b) 144 (c) 72

2.17. (6!) = 720; maka (7!) = (7)(720) = 5040 dan (8!) = (8)(5040) = 40.320

2.19. n = 23

2.21. (a) (4!)(5!)(6!) (b) (3!)(4!)(5!)(6!)

2.23. 5040

2.25. 86.400

2.27. (a) (10!)/(4!)(4!)(2!) (b) (12!)/(5!)(4!)(3!)

2.31. (a) P(11, 9) (b) P(11; 2, 3, 4)

2.33. (a) 512 (b) 84 (c) 36

2.35. Ada 504 cara.

2.37. C(n, r) • (r – 1)!

2.39. Banyaknya cara adalah r dimana r = C(14, 8) • (7!) • (6!).

2.41. P(7; 4, 2, 1) • C(8, 4)

2.47. P(10; 2, 3, 4, 1) • C(23, 1)

2.49. (a) C(10, 6) • C(12, 6) (b) C(12, 7) • C(10, 5) + C(12, 8) • C(10, 4) + C(12 , 9) • C(10, 3) + C(12, 10) • C(10, 2)

+ C(12, 11) • C(10, 1) + C(12, 12) • C(10, 0)

1.57. (a) (18!)/[(4!) • (1!) • (4!)4 • (2!)'l (b) (18!)/[(2) • (2) • (5 !)2 • (4!) • (2!)2] (c) C(18; 7, 6, 5)

2.58. (a) 840 (b) 74 (c) 79

2.61. C(28, 4)

2.63. (a) C(14, 4) (b) C(9, 4) (c) C(8, 3) + C(6, 3) + C(4, 3)

2.65. C(10, 4)

2.69. C((r – p) + n – 1, n – 1), di mana p = p1 + p2 + • • • + pn

2.72. C(15, 5) – C(8, 5)

2.74. (a) 25 (b) 27

2.78. 275

2.80. (a) Misalkan X = {1, 2, . . . , n} dan A adalah himpunan bilangan di X yang tidak bebas kuadrat. Banyaknya bilangan bulat bebas kuadrat dalam X jelas adalah n – N(A). Untuk menghitung N(A), lakukan sebagai berikut. Misalkan P = {p1, p2, . . . , pr}, di mana setiap pi adalah bilangan prima yang tidak melebihi akar kuadrat dari n. Misalkan Ai, himpunan bilangan-bilangan dalam X yang habis dibagi kuadrat pi Hitung Si (i = 1, 2, . . , r) seperti pada Teorema 1.6.1. Maka N(A) sama dengan S1 S2 + . . . + (– 1)r−1. (b) 100 (42

3 + 0 0)

2.82. (a) Banyaknya permutasi = 6! = 720, jumlah kekacauan = D6 = 265 (b) 265/720 (c) [C(6, 1) • D5]/(6!) = 0,366667 (d) 1 – 0,366667 (e) [C(6, 2) • D4]/(6!) (f) 1/720

2.84. Jawabannya adalah 0.

2.86. Ada 120 cara.

BAB 3

3.1. (a) 1 + x + x2 + x3 (b) x4 + x5 + x6 + • • • (c) 1 + x + x2 + x3 + • • • (d) 1 – x + x2 – x} + x4 – • • •

3.3. (a) {16, 32, 24, 8, 1, 0, 0, 0, . . .} (b) {1,1, 3/2, 1/3!, 1/4!, 1/5!, . . .} (c) {0, 0, 0, 1, 1, 1,1 }

3.5. C(15, 8)

3.7. (x + x2 +x3 + + x’)4

3.9. C(18, 3) – 4 • C(12, 3) + 6 • C(6, 3)

3.11. C(12, 2) – 3 • C(6, 2)

3.13. Ada 18 cara.

3.17. Koefisien pangkat kesepuluh x dalam f(x), di mana f(x) = (x + x2)(x + x2 + x3)(x + x2 +)2

3.19. Fungsinya adalah x6 (1 – x4)3 • (1 – x)–4.

3.21. 10

3.23. 6

3.25. 30

3.27. (x4 + x8 + • • •)(1 + x3 + x6 + • • •)(1 + x2 + x4 + • • •)(1 + x + x2 +)

3.29. C(n – r + 1, r)

3.37. Banyaknya bilangan tersebut adalah t, dimana t = (1/4)(2r + 2r) dimana r genap. Jika r ganjil, maka t pasti nol.

3 41. (mis + e–x)/2 – l](mis – 1)4

BAB 4

4 1. f(n) = f(n – 1) + n dimana f(1) = 2; f(9) = 46

4 3. f(n) = 2f(n – 1) + 1 dengan f(1) = l; f(n) = 2" – 1

4 5. f(n) = f(n – 1) + f(n – 2) dengan f(1) = 2 dan f(2) = 3

4 8. f(n) = 2f(n – 1) dengan f(1) = 2

4.10. (a) k = 2 (b) Kondisi awal tidak berurutan.

4.12. f(n) = 1 + n + 2n

4.14. Polinomial karakteristik adalah (x – 1) (x –2)2 (x – 3) dan f(n) = A + B • 2n + C • n • 2n + D • 3n adalah solusi umum dari relasi f(n + 4) = 8f(n + 3) – 23f(n + 2) + 28f(n + 1) – 12f(n).

4.16. g(n) = A(–1)n + B(m – 1)n, di mana A = (–1)/m dan B = 1/m dan

f(n) = (m – 1)g(n – 1)

4.18. f(n) = A + B • (3)n – 8n; dimana A = 1 dan B = 3

4.20. A(4)n + 5n(4)n

4.22. A(2)n + B • n • (2)n + (1/2) • n2 • (2)n

4 24. p = –5, q = 6, dan r = 8

4 26. f(n) = koefisien xn dalam g(x) = 2/(1 + x) – 1/(1 – x)2+ 2/(1 – x)3

4.28. f(n) = 5n2 – 4n

4.30. f(n) = d + c log n

4.32. Relasinya adalah g(n) = 7g(n/2) + 18(n2)2 dengan g(1) = 0. Solusinya adalah 6 • nr – 6 • n2 di mana r = log 7.

4.34. (a) f(n) = 2f(n – 1) dengan f(1) = 0 (b) f(n) = f(n/2) + (n – 1) dengan f(1) = 0 (c) Kedua solusi tersebut adalah (1) n(n – 1)/2 dan (2) 2n – log n – 2. Ketika n > 3, yang kedua lebih efisien daripada yang pertama.

BAB 5

5.1. P = {1, 3}

5.3. (b) Dimungkinkan untuk menggambar K4 sedemikian rupa sehingga tidak ada sisi yang berpotongan. Tidak mungkin melakukannya untuk K5.

5.5. Gambarlah grafik sederhana seperti yang disarankan. Ada batas antara setiap pasangan kota kecuali antara Boston dan Moskow, dan antara Boston dan Praha.

5.7. (a) Dua (b) Dua

5.9. pq

5.11. Misalkan busur digraf adalah (1, 2), (1, 3), (1, 5), (2, 3), (3, 4), (3, 5), dan (4, 5) .

(a) Matriks ketetanggaan A = (aij) adalah matriks 5 × 5 di mana a12 = a13 = a15 = a23 = a34 = a35 = a45 = 1 dan semua elemen lainnya adalah nol.

(b) (derajat luar simpul 1) = 0, (derajat luar simpul 1) = 3 (derajat luar simpul 2) = 1, (derajat luar simpul 2) = 1 (derajat luar simpul 3) = 2, (derajat luar simpul 3) = 2 (Derajat luar simpul 4) = 1, (derajat luar simpul 4) = 1 (derajat keluar simpul 5) = 3, (derajat luar simpul 5) = 0

(c) (1, 2) (1, 3) (1, 5) (2, 3) (3, 4) (3, 5) (4, 5)

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{bmatrix} -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

5.13. Grafik tersebut terlihat seperti gelang atau cincin yang ditaburi batu sedemikian rupa sehingga setiap simpul diwakili oleh sebuah batu. Graf dengan n simpul tersebut dilambangkan dengan Zn dan disebut “lubang ganjil” jika n ganjil dan lebih dari 3. Disebut “lubang genap” jika n genap dan lebih dari 3.

5.15. (a) G = (V, E), di mana V = {1, 2, 3, 4} dan E = {{1, 2}, {3, 4}}

(b) G = (V, E), di mana V = {1, 2, 3, 4} dan E = {{1, 2}, {2, 3}, {3, 4}, {4, 1} }

(c) (nr)/2, jadi paling sedikit salah satu dari kedua bilangan tersebut genap.

5.19. (a) 1 → 2 → 3 → 4 → 5 → 2 → 6 (b) 1 → 2 → 3 → 4 → 5 → 6 (c) 2 → 3 → 4 → 5 → 2 (d) Satu (e) { 1}, {6}, {2, 3, 4, 5} (f) Ini adalah matriks 6 × 6 di mana (1) semua elemen pada baris 6 adalah 0, (2) semua elemen pada kolom 1 adalah 0 kecuali elemen pertama, dan (3) semua elemen yang tersisa adalah 1.

5.21. G = (V, E), dimana V = {1, 2, 3, 4, 5} dan E = {{1, 2}, {2, 3}, {3, 4}, {4, 2}, {2, 5}, {5, 1}}

5.23. Jika graf terhubung, setiap elemen adalah 1. Secara umum, jika G memiliki n simpul dan k komponen, matriks keterjangkauan n × n A dari G akan memiliki k submatriks sepanjang diagonal A sehingga setiap elemen dalam setiap submatriks adalah 1 dan setiap elemen lain di A adalah nol. Jika Gi adalah komponen dari G dengan simpul ni, submatriks yang sesuai dengan komponen ini akan menjadi matriks ni × ni.

5.27. G tidak terhubung.

5.29. Tiga simpul di atas ditandai 1, 2, dan 3. Tiga simpul bertanda 8, 9, dan 4 dari kiri ke kanan. Tiga simpul di bawah ditandai 7, 6, dan 5 dari kiri ke kanan. Busur (i, j) di mana i < j adalah (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), dan (8, 9). Busur yang tersisa adalah (i, j) di mana i > j.

BAB 6

6.3. Akan ada jalur berarah dari setiap simpul ke setiap simpul.

6.5. Tidak

6.7. Kata dimulai dengan C karena jumlah barisnya 2 sama dengan jumlah kolomnya ditambah 1. Kata diakhiri dengan B karena jumlah kolomnya sama dengan jumlah barisnya ditambah 1. Untuk dua huruf lainnya, jumlah baris sama dengan jumlah kolom. Jumlah baris A dan D adalah 2 dan 3. Jadi frekuensi A, B, C, dan D berturut-turut adalah 2, 2, 2, dan 3. Gambarlah digraf dengan 4 simpul A, B, C, D. Gambarlah an dari huruf X (X adalah salah satu dari empat huruf ini) ke huruf Y (Y juga salah satu dari empat huruf ini, huruf X dan Y perlu tidak berbeda) jika elemen dalam matriks yang bersesuaian dengan baris X dan kolom Y adalah 1. Dalam digraf yang dihasilkan akan ada jalur Euler berarah (tidak harus unik) dari C ke B. Setiap jalur tersebut akan memberikan sebuah kata.

6.9. (a) Digrafnya adalah G = (V, A) di mana V = {0, 1, 2} dan A adalah hasil kali Cartesian V × V. Busur dari titik i ke titik j diberi kata ij. Busur berurutan pada barisan berikut akan menghasilkan rangkaian Euler yang dimulai dari simpul 0 dan berakhir di simpul 0: < 00 01 11 12 22 21 10 02 21 >. Huruf pertama dari sembilan kata ini mendefinisikan de Bruijn barisan B(3, 2) = < a1 a2 a3 a4 a5 a6 a7 a8 a9 > = < 0 0 1 1 2 2 1 0 2 > dan setiap kata dua huruf yang menggunakan tiga simbol 0, 1, dan 2 adalah bentuk ai ai + 1, di mana i adalah sembarang bilangan bulat sehingga 1 < i < 9 dan penambahan subskripnya adalah modulo 9. 5.11. V = {1, 2, 3, 4}; E = {{1, 2}, {2, 3}, {3, 4}, {4, 1}, {1, 3}}

6.13. V = {1, 2, 3, 4}; E = {{1, 2}, {1, 3}, {1, 4}}

6.15. Banyaknya sisi dalam siklus Hamilton dari graf dengan n simpul adalah n. Banyaknya rusuk pada sembarang siklus dari graf bipartit adalah genap. Jadi graf bipartit dengan jumlah simpul ganjil tidak bisa menjadi Hamiltonian.

6.17. Ya, menurut definisi. Kebalikannya tidak benar, untuk mempertimbangkan contoh tandingan dengan V = {1, 2, 3} dan A = {(1, 2), (2, 3)}.

BAB 7

7.1. m = n – k

7.3. 19

7.5. 24

7.7. Ini adalah sebuah jembatan.

7.9. Tidak perlu. Jika himpunan simpul dari graf lengkap dengan lima simpul adalah {1, 2, 3, 4, 5}, maka T dan T' adalah dua pohon merentang berbeda dengan E = {{1, 2}, {2, 3}, {3, 4}, {4, 5}} dan E' = {{1, 3}, {3, 5}, {2, 5}, {2, 4}}.

7.13. Pohonnya adalah T = (V, E), dimana V = {1, 2, 3, 4, 5, 6, 7, 8, 9} dan E = {(1, 8), (2, 8), (3, 7), (4, 7), (5, 7), (7, 6), (8,

6), (6, 9)}.

7.15. OMONG KOSONG

7.17. A = 110, B = 00, C = 1110, D = 1111, E = 10, R = 01. Kata tersebut adalah 110 00 01 110 1110 110 1111 110 00 01 110 Panjangnya paling banyak 31.

7.19. (a) n + 1 = 14 dan 3 < log 14 < 4, jadi m = 3. Jadi tingginya tidak boleh kurang dari 3. (b) Lantai 13/2 adalah 6, jadi tingginya tidak lebih dari 6 .

7.21. Pohon berakar pada H. Pohon kiri memiliki G, B, A, dan C. Pohon kanan memiliki P, R, dan Y. Anak kiri G adalah B. Tidak ada anak kanan untuk G. Anak kiri dari fl adalah A dan anak kanan dari B adalah C. Pada subpohon kanan yang berakar pada H, tidak ada simpul yang memiliki anak kiri.

BAB 8

8.5. Sisi-sisinya adalah {1, 5}, {1, 4}, {3, 4}, {2, 6}, dan {5, 6} . Berat pohon adalah 31.

BAB 9

9.1.

$$A^{(7)} = \begin{bmatrix} 0 & 3 & 2 & 2 & 3 & 4 & 2 \\ - & 0 & -1 & -1 & 2 & 1 & 0 \\ - & 1 & 0 & 0 & 3 & 2 & 1 \\ - & 1 & 0 & 0 & 3 & 2 & 1 \\ - & 0 & -1 & -1 & 0 & 1 & -1 \\ - & 1 & 0 & 0 & 3 & 2 & 0 \end{bmatrix} \quad P^{(7)} = \begin{bmatrix} 1 & 5 & 5 & 5 & 5 & 5 & 5 \\ 1 & 2 & 3 & 4 & 5 & 3 & 7 \\ 1 & 6 & 3 & 6 & 5 & 6 & 7 \\ 1 & 3 & 3 & 4 & 3 & 3 & 3 \\ 1 & 2 & 2 & 2 & 2 & 6 & 2 \\ 1 & 6 & 6 & 6 & 6 & 6 & 7 \end{bmatrix}$$

9.3. Busur pohon yang berakar pada simpul 1 adalah (1, 5), (5, 7), (7, 6), (5, 2), (2, 3), dan (3, 4).

9.5. Dalam A(4) elemen yang bersesuaian dengan baris keempat dan kolom kedua adalah 4. Jadi S.D. dari 4 ke 2 tanpa menyentuh 5, 6, atau 7 akan menjadi 4. Selanjutnya, pada p(4), elemen yang sesuai dengan baris keempat dan kolom kedua adalah 2, menunjukkan bahwa kita langsung dari 4 ke 2.

9.7. Ganti traktor pada akhir tahun pertama. Total biayanya adalah 12 + 32 = 44.

DAFTAR PUSTAKA

AHO, A. V., HOPCROFT, J. E., and ULLMAN, J. D. Data Structures and Algorithms, Addison-Wesley, Reading, Mass., 1983.

AIGNER, M. Combinatorial Theory, Springer-Verlag, New York, 1979.

ANDERSON, I. A First Course in Combinatorial Mathematics, Oxford University Press, Oxford, 1979.

APPEL, K., and HAKEN, W. "Every Planar Map Is Four Colorable," Bull. Amer. Math. Soc. 82 (1976), 711–712.

BAASE, S. Computer Algorithms: Introduction to Design and Analysis, Addison-Wesley, Reading, Mass., 1978.

BEHZAD, M., CHATRAND, G., and LESINAK-FOSTER, L. Graphs and Digraphs, Wads-worth, Belmont, Calif., 1979.

BELLMORE, M., and NEMHAUSER, G. L. "The Traveling Salesman Problem: A Survey," Oper. Res. 16 (1968), 538–558.

BERGE, C. The Theory of Graphs and Its Applications, Wiley, New York, 1962.

BIRKHOFF, G. D., and LEWIS, D. C. "Chromatic Polynomials," Trans. Amer. Math. Soc. 60 (1960), 355–451.

BONDY, J. A., and MURTY, U. S. R. Graph Theory with Applications, Elsevier, New York, 1976.

BOYER, C. B. History of Mathematics, Wiley, New York, 1968.

BUSSEY, W. H. "Origins of Mathematical Induction," American Math. Monthly 24(1917), 199–207.

CARRE, B. Graphs and Networks, Clarendon Press, Oxford, 1979.

CHANG, S. K. "The Generation of Minimal Trees in a Steiner Topology," J. Assoc. Comput. Mach. 19 (1972), 699–711.

CHARTRAND, G. Graphs as Mathematical Models, Wadsworth, Belmont, Calif., 1977.

CHARTRAND, G., KAPOOR, S. F., and KRONK, H. V. "A Generalization of Hamiltonian-Connected Graphs," J. Math. Pures Appl. (9) 48 (1969), 109–116.

CHERITON, D., and TARJAN, R. E. "Finding Minimum Spanning Trees," SIAM. J. Comput. 5 (1976), 724–742.

COHEN, D. I. Basic Techniques of Combinatorial Theory, Wiley, New York, 1978.

COOK, S. A. “The Complexity of Theorem-Proving Procedures,” in Proceedings Third ACM Symposium on the Theory of Computing, Assoc. for Computing Machinery, New York, 1971, pp. 151–158.

DE BRUIJN, N. G. “A Combinatorial Problem,” Nederl. Akad. Wetensch. Proc. 49 (1946), 758–764.

DEO, N. Graph Theory with Applications to Engineering and Computer Science, Prentice-Hall, Englewood Cliffs, N.J., 1974.

DIJKSTRA, E. W. “A Note on Two Problems in Connection with Graphs,” Numer. Math. 1 (1959), 269–271.

DIRAC, G. A. “Some Theorems on Abstract Graphs,” Proc. London Math. Soc. 2 (1952), 69–81.

DREYFUS, S. E. “An Appraisal of Some Shortest-Path Algorithms,” Oper. Res. 17 (1969), 395–112.

EDMONDS, J. “Paths, Trees and Flowers,” Canad. J. Math. 17 (1965b), 449–467.

EVEN, S. Graph Algorithms, Computer Science Press, Potomac, Md., 1979.

FLOYD, R. W. “Algorithm 97: Shortest Path,” Comm. ACM 7 (1964), 345.

GABOW, H. P., GALIL, Z., SPENCER, T., and TARJAN, R. E. “Efficient Algorithms for Finding Minimum Spanning Trees in Undirected and Directed Graphs,” Combinatorica 6 (1986), 109–112.

GAREY, M. R., and JOHNSON, D. S. Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, San Francisco, 1979.

GIBBONS, A. Algorithmic Graph Theory, Cambridge University Press, Cambridge, 1985.

GOLDBERG, S. Introduction to Difference Equations, Wiley, New York, 1958.

GOLDMAN, A. J. “Discrete Mathematics in Government,” Lecture on the Applications of Discrete Mathematics, SIAM, Troy, N.Y. (1982).

GOLOMB, S. W. Shift Register Sequences, Holden-Day, San Francisco, 1967.

GOLOVINA, L. I., and YAGLOM, l. M. Induction in Geometry, D. C. Heath, Boston, 1963.

GONDRAN, M., and MINOUX, M. Graphs and Algorithms, Wiley, New York, 1984.

GOULD, R. Graph Theory, Benjamin-Cummings, Menlo Park, Calif., 1988.

GRAHAM R. L., and HELL, P. “On the History of the Minimum Spanning Tree Problem,” Bell Lab. Rep. (1982).

GRAHAM, R. L., ROTHSCHILD, B. L., and SPENCER, J. H. Ramsey Theory, Wiley, New York, 1980.

GRIMALDI, R. P. Discrete and Combinatorial Mathematics, Addison-Wesley, Reading, Mass., 1985.

HAKEN, W. “An Attempt to Understand the Four Color Problem,” J. Graph Theory 1 (1977), 193–206.

HALMOS, P. Naive Set Theory, Van Nostrand, Princeton, N.J., 1960.

HARARY, F. “The Four Color Conjecture and Other Graphical Diseases,” in Proof Techniques in Graph Theory, Academic Press, New York, 1969b.

HARARY, F. Graph Theory, Addison-Wesley, Reading, Mass., 1969a.

HENKIN, L. “On Mathematical Induction,” Amer. Math. Monthly 67 (1960), 323–337.

HILLIER, F. S., and LIEBERMAN, G. J. Introduction to Operations Research, 4th ed., Holden-Day, Oakland, Calif., 1986.

HU, T. C. Combinatorial Algorithms, Addison-Wesley, Reading, Mass., 1982.

HUFFMAN, D. A. “A Method for the Construction of Minimum Redundancy Codes,” Proc. IRE 40 (1952), 1098–1101.

HUTCHINSON, J. P., and WILF, H. S. “On Eulerian Circuits and Words with Prescribed Adjacency Patterns,” J. Combin. Theory A18 (1975), 80–87.

KARMARKAR, N. “A New Polynomial Algorithm for Linear Programming,” Combinatorica 4 (1984), 373–395.

KARP, R. M. “A Simple Derivation of Edmonds’ Algorithm for Optimum Branchings,” Networks 1 (1972b), 265–272.

KARP, R. M. “Combinatorics, Complexity and Randomness,” Comm. ACM 29 (1986), 98–111.

KARP, R. M. “Reducibility among Combinatorial Problems,” in Complexity of Computer Computations, Plenum Press, New York, 1972a.

KHACHIYAN, L. G. “A Polynomial Algorithm in Linear Programming” (in Russian); English translation in Soviet Math. Dokl. 20 (1979), 191194.

KNUTH, D. E. The Art of Computer Programming, Vol. 1, Addison-Wesley, Reading, Mass., 1973a.

KNUTH, D. E. The Art of Computer Programming, Vol. 3, Addison-Wesley, Reading, Mass., 1973b.

KRISHNAMURTHY, V. Combinatorics: Theory and Applications, Ellis Horwood, Chichester, West Sussex, England, 1986.

KRUSKAL, J. B. “On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem,” Proc. Amer. Math. Soc. 1 (1956), 48–50.

KWAN, M. K. “Graphic Programming Using Odd or Even Points,” Chinese J. Math. 1 (1962), 273–277.

LAWLER, E. L. Combinatorial Optimization: Networks and Matroids, Holt, Rinehart and Winston, New York, 1976.

LAWLER, E. L., LENSTRA, J. K., RINNOOY KAN, A. H. G., and SHMOYS, D. B. The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, Wiley, New York, 1985.

LEVY, H., and LESSMAN, F. Finite Difference Equations, Macmillan, New York, 1961.

LEWIS, H. R., and PAPADIMITRIOU, C. H. Elements of the Theory of Computation, Prentice-Hall, Englewood Cliffs, N.J., 1981.

LICK, D. R. “A Sufficient Condition for Hamiltonian Connectedness,” J. Comb. Theory 8 (1970), 444–445.

LIU, C. L. Elements of Discrete Mathematics, McGraw-Hill, New York, 1985.

LIU, C. L. Introduction to Combinatorial Mathematics, McGraw-Hill, New York, 1968.

MACMAHON, P. Combinatory Analysis, Vol. 1 (1915) and Vol. 2 (1916); reprinted in one volume by Chelsea, New York, 1960.

MARKOWSKY, G. “Best Huffman Trees,” Acta Inform. 16 (1981), 363–370.

MAY, K. O. “The Origin of the Four Color Conjecture,” Isis 56 (1965), 346–348.

MINIEKA, E. Optimization Algorithms for Networks and Graphs, Marcel Dekker, New York, 1978.

MOON, J. W. Topics in Tournaments, Holt, Rinehart and Winston, New York, 1968.

NEMHAUSER, G. L. “A Generalized Label Setting Algorithm for the Shortest Path between Specified Nodes,” J. Math. Anal. Appl. 38 (1972), 328–334.

ORE, O. Graphs and Their Uses, New York, 1963.Random House, N.Y.

PAPADIMITRIOU, C. H. “The Complexity of Edge Traversing,” J. Assoc. Comput. Mach. 23 (1976), 544–554.

PAPADIMITRIOU, C. H. and STEIGLITZ, K. Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Englewood Cliffs, N.J., 1982.

POLYA, G. Induction and Analogy in Mathematics, Princeton Univ. Press, Princeton, N.J., 1963.

PRATT, V. “Every Prime Has a Succinct Certificate,” SIAM J. Comput. 4 (1975), 214–220.

PRIM, R. C. “Shortest Connection Networks and Some Generalizations,” Bell System Tech. J. 36 (1957), 1389–1401.

RALSTON, A. “De Bruijn Sequences—A Model Example of the Interaction of Discrete Mathematics and Computer Science,” Math. Mag. 55 (1982), 131–143.

READ, R. C. “An Introduction to Chromatic Polynomials,” J. Combin. Theory 4 (1968), 52–71.

REDEI, L. “Ein kombinatorischer Satz,” Acta Litt. Sci. Szegu 1 (1934), 39–43.

REINGOLD, E. M., NIEVERGELT, J., and DEO, N. Combinatorial Algorithms: Theory and Practice, Prentice-Hall, Englewood Cliffs, N.J., 1977.

RIORDAN, J. An Introduction to Combinatorial Analysis, Princeton University Press, Princeton, N.J., 1958.

ROBBINS, H. E. “A Theorem on Graphs with an Application to a Problem of Traffic Control,” Amer. Math. Monthly 46 (1939), 281–283.

ROBERTS, F. S. Applied Combinatorics, Prentice-Hall, Englewood Cliffs, N.J., 1984

ROBERTS, F. S. Discrete Mathematical Models with Applications to Social Biological and Environmental Problems, Prentice-Hall, Englewood Cliffs, N.J., 1976.

ROBERTS, F. S. Graph Theory and Its Applications to Problems of Society, SIAM, Philadelphia, 1978.

RONSE, C. Feedback Shift-Registers, Springer-Verlag, New York, 1982.

RYSER, H. J. Combinatorial Mathematics, Cams Mathematical Monographs No. 14, Mathematical Association of America, Washington D.C., 1963.

SCHRIJVER, A. Theory of Linear and Integer Programming, Wiley, New York, 1986.

SOMINSKIT, I. S. The Method of Math Induction, D. C. Heath, Boston, 1963.

STANAT, D. F., and MCALLISTER, D. F. Discrete Mathematics in Computer Science, Prentice-Hall, Englewood Cliffs, N.J., 1977.

STANDISH, T. A. Data Structure Techniques, Addison-Wesley, Reading, Mass., 1980.

STOLL, R. R. Set Theory and Logic, W. H. Freeman, San Francisco, 1963.

SWAMY, M. N. S., and THULASIRAMAN, K. Graphs, Networks and Algorithms, Wiley, New York, 1981.

SYSLO, M. M., DEO, N., and KOWALIK, J. S. Discrete Optimization Algorithms with Pascal Structures, Prentice-Hall, Englewood Cliffs, N.J., 1983.

TABOURIER, Y. “All Shortest Distances in a Graph,” Discrete Math. 4 (1973), 83–87.

TARJAN, R. E. “Depth First Search and Linear Graph Algorithms,” SIAM J. Comput. 1 (1971), 146–160.

TOWNSEND, M. Discrete Mathematics: Applied Combinatorics and Graph Theory, Benjamin-Cummings, Menlo Park, Calif., 1987.

TUCKER, A. Applied Combinatorics, 2nd ed. Wiley, New York, 1984.

TYMOCZKO, T. “Computers, Proofs and Mathematicians: A Philosophical Investigation of the Four Color Proof,” Math. Mag. 53 (1980), 131138.

WARSHALL, S. “A Theorem on Boolean Matrices,” J. Assoc. Comput. Mach. 9 (1962), 11–12.

WHITWORTH , W. A. Choice and Chance (reprint of the fifth edition, 1901), Hafner Press, New York, 1965.

WILF, H. S. Algorithms and Complexity, Prentice-Hall, Englewood Cliffs, N.J., 1986.

WILSON, R. J. Introduction to Graph Theory, 2nd ed. Longman Group, Harlow, Essex, England, 1979.

YEMELICHEV, V. A., KOVALEV, M. M., and KRAVTSOV, M. K. Polytopes, Graphs and Optimisation, Cambridge University Press, Cambridge, 1984.

MATEMATIKA DISKRIT

Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

BIO DATA PENULIS

Penulis memiliki berbagai disiplin ilmu yang diperoleh dari Universitas Diponegoro (UNDIP) Semarang. dan dari Universitas Kristen Satya Wacana (UKSW) Salatiga. Disiplin ilmu itu antara lain teknik elektro, komputer, manajemen dan ilmu sosiologi. Penulis memiliki pengalaman kerja pada industri elektronik dan sertifikasi keahlian dalam bidang Jaringan Internet, Telekomunikasi, Artificial Intelligence, Internet Of Things (IoT), Augmented Reality (AR), Technopreneurship, Internet Marketing dan bidang pengolahan dan analisa data (komputer statistik).

Penulis adalah pendiri dari Universitas Sains dan Teknologi Komputer (Universitas STEKOM) dan juga seorang dosen yang memiliki Jabatan Fungsional Akademik Lektor Kepala (Associate Professor) yang telah menghasilkan puluhan Buku Ajar ber ISBN, HAKI dari beberapa karya cipta dan Hak Paten pada produk IPTEK. Penulis juga terlibat dalam berbagai organisasi profesi dan industri yang terkait dengan dunia usaha dan industri, khususnya dalam pengembangan sumber daya manusia yang unggul untuk memenuhi kebutuhan dunia kerja secara nyata.

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
JL. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

MATEMATIKA DISKRIT

PENERBIT :

YAYASAN PRIMA AGUS TEKNIK

JL. Majapahit No. 605 Semarang

Telp. (024) 6723456. Fax. 024-6710144

Email : penerbit_ypat@stekom.ac.id

YAYASAN PRIMA AGUS TEKNIK