Piagam Penghargaan

No. 2035/SFP/VI/24

PIAGAM PENGHARGAAN INI DIBERIKAN KEPADA:

Dr. Retno Tri Nalarsi ST., MT.

---

yang telah menulis dan menerbitkan buku dengan judul *Logika Matematika: Teori dan Praktik* di Penerbit Haura Utama. Buku tersebut terdaftar di Perpustakaan Nasional Republik Indonesia dengan nomor ISBN: **978-623-492-884-6** dan nomor anggota IKAPI Penerbit Haura 375/JBA/2020

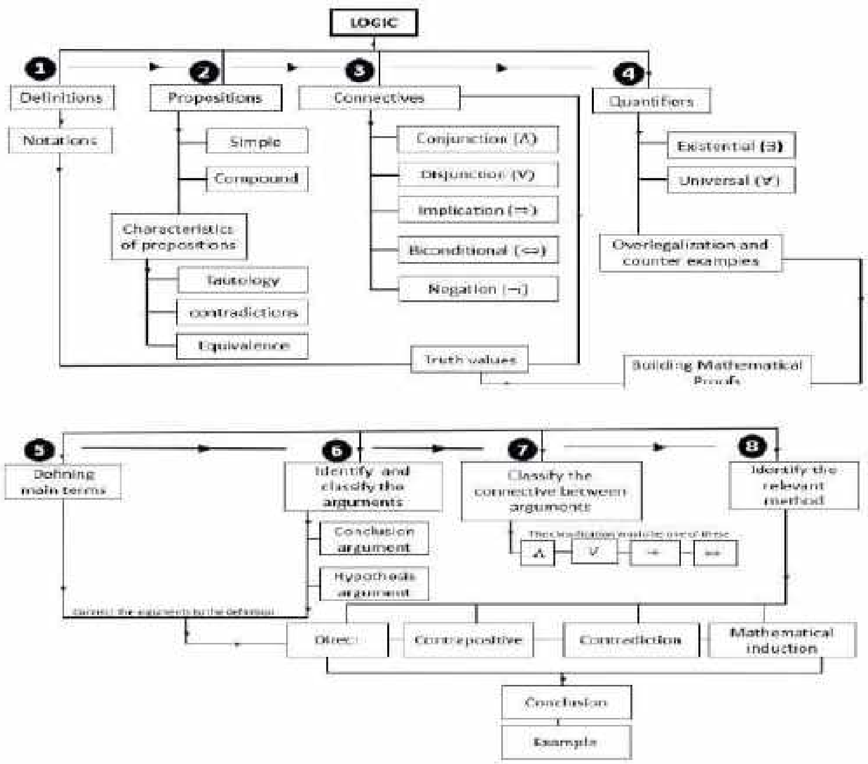
Cece Abdulwaly
CEO Haura Utama

logmat2

by jurnalkomunika22 1

1 PENGANTAR, PEMAHAMAN DAN KONSEP LOGIKA

Peta Konsep



Gambar 1. Peta Konsep Logika dan Bukti Matematis (Issa & Majuma, 2019)

Pendahuluan

Matematika adalah ilmu pasti. Setiap pernyataan matematika harus tepat. Oleh karena itu, harus ada penalaran yang tepat dalam setiap bukti matematis. Penalaran yang tepat melibatkan logika. Belajar logika membantu dalam meningkatkan kemampuan seseorang dalam penalaran sistematis dan logis. Ini juga mengembangkan keterampilan memahami berbagai pernyataan dan validitasnya. Logika memiliki aplikasi skala luas dalam desain sirkuit, pemrograman komputer. Oleh karena itu, studi logika menjadi penting.

Pembahasan

Pernyataan dan nilai kebenarannya

Ada berbagai sarana komunikasi, yaitu lisan dan tulisan. Sebagian besar komunikasi melibatkan penggunaan bahasa di mana ide-ide disampaikan melalui kalimat.

Ada berbagai jenis kalimat seperti:

1. Deklaratif (Asertif)→ ringkas dan jelas

2. Imperatif (Perintah atau permintaan)

3. Seruan (Emosi, kegembiraan)

4. Interogatif (Pertanyaan)

Pernyataan

Pernyataan adalah kalimat deklaratif yang benar atau salah, tetapi tidak keduanya sekaligus. Pernyataan dilambangkan dengan huruf p, q, r, ...

Misalnya:

a. 3 adalah angka ganjil
b. 5 adalah persegi yang sempurna
c. Matahari terbit di timur
d. $x + 3 = 6$, ketika $x = 3$

Nilai Kebenaran

Sebuah pernyataan bisa Benar atau Salah. Nilai Kebenaran dari pernyataan "benar" didefinisikan sebagai T (*TRUE*) dan pernyataan "salah" didefinisikan sebagai F (*FALSE*)

Catatan: 0 dan 1 juga dapat digunakan untuk T dan F masing-masing

Pertimbangkan pernyataan berikut:

a. Tidak ada bilangan prima antara 23 dan 29
b. Matahari terbit dari barat
c. Kuadrat bilangan real adalah negatif
d. Jumlah sudut segitiga bidang adalah 180°

Di sini nilai kebenaran pernyataan 1 dan 4 adalah T dan nilai kebenaran pernyataan 2 dan 3 adalah F.

Catatan: Kalimat seperti seruan, interogatif (pertanyaan), imperatif (perintah/permintaan), tidak dianggap sebagai pernyataan karena nilai kebenaran pernyataan tidak dapat ditentukan.

Kalimat terbuka

Kalimat terbuka adalah kalimat yang kebenarannya dapat bervariasi sesuai dengan beberapa keadaan, yang tidak disebutkan dalam kalimat.

Catatan: Kalimat terbuka tidak dianggap sebagai pernyataan logis.

Misalnya:

a. $x \times 5 = 20$
   Ini adalah kalimat terbuka karena kebenarannya tergantung pada nilai x (jika x = 4 benar, dan jika $x \neq 4$ salah).
b. Makanan Cina sangat enak.
   Ini adalah kalimat terbuka karena kebenaran bervariasi dari individu ke individu.

LATIHAN 1

Nyatakan yang mana dari kalimat berikut yang merupakan pernyataan. Jika ada pernyataan, tuliskan nilai kebenarannya.

1. Matahari adalah bintang.
2. Semoga Tuhan memberkati Anda!
3. Jumlah sudut interior segitiga dalam ruang Euclidean adalah 180°.
4. Setiap bilangan real adalah bilangan kompleks.
5. Mengapa Anda kesal?
6. Setiap persamaan kuadrat memiliki dua akar nyata.
7. $\sqrt{-9}$ adalah bilangan rasional.
8. $x^2 - 3x + 2 = 0$, dimana bahwa x = -1 atau x = -2.
9. Tolong ambilkan saya segelas air.
10. Dia orang yang baik.
11. Dua adalah satu-satunya bilangan prima genap.
12. Sungguh pemandangan yang mengerikan!
13. Jangan ganggu.
14. $x^2 - 3x - 4 = 0, x = -1$.
15. Bisakah Anda berbicara bahasa Prancis?
16. Kuadrat dari bilangan real apa pun adalah positif.
17. Warnanya merah.
18. Setiap jajaran genjang adalah belah ketupat.

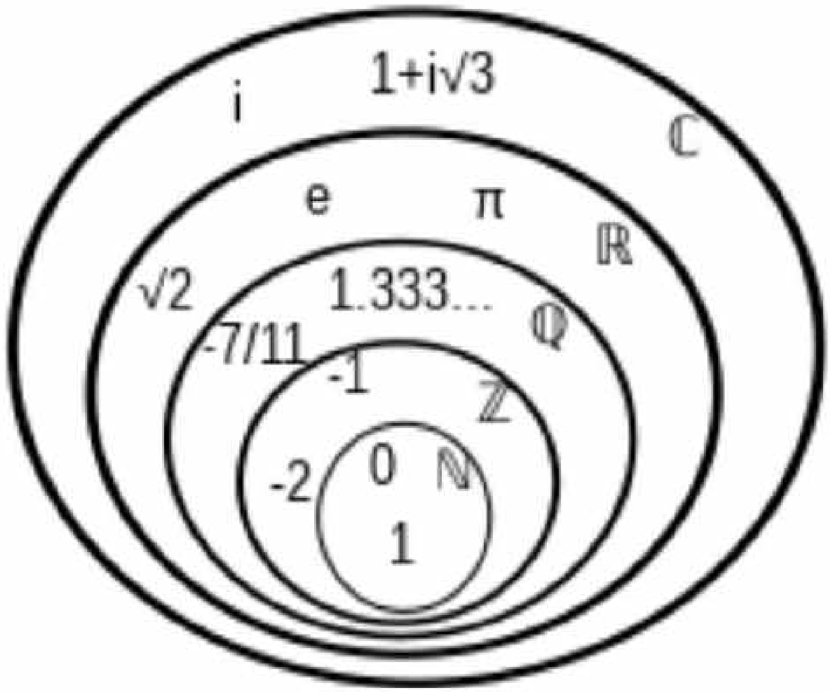
Penyelesaian:

1. Matahari adalah bintang. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
2. Semoga Tuhan memberkati Anda! (Ini adalah tanda seru; maka ini bukan pernyataan).
3. Jumlah sudut interior segitiga adalah 180°. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
4. Setiap bilangan real adalah bilangan kompleks. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
5. Mengapa Anda kesal? (Ini adalah kalimat interogatif; maka itu bukan pernyataan).
6. Setiap persamaan kuadrat memiliki dua akar nyata. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").
7. $\sqrt{-9}$ adalah bilangan rasional. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").

8. $x^2 - 3x + 2 = 0$, mengisyaratkan bahwa x = -1 atau x = -2. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").
9. Tolong ambilkan saya segelas air. (Ini adalah kalimat penting; karenanya ini bukan pernyataan).
10. Dia orang yang baik. (Ini adalah kalimat terbuka; maka ini bukan pernyataan).
11. Dua adalah satu-satunya bilangan prima genap. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
12. Sungguh pemandangan yang mengerikan! (Ini adalah tanda seru; maka ini bukan pernyataan).
13. Jangan ganggu. (Ini adalah kalimat penting; karenanya ini bukan pernyataan).
14. $x^2 - 3x - 4 = 0, x = -1$. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
15. Bisakah Anda berbicara bahasa Prancis? (Ini adalah kalimat interogatif; maka ini bukan pernyataan).
16. Kuadrat dari bilangan real apa pun adalah positif. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F"); karena, 0 adalah bilangan real dan kuadrat dari 0 adalah 0 yang tidak positif atau negatif.
17. Warnanya merah. (Ini adalah kalimat terbuka; karenanya ini bukan pernyataan); kebenaran kalimat ini tergantung pada referensi ke kata ganti "itu".
18. Setiap jajaran genjang adalah belah ketupat. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").

2 REPRESENTASI ANGKA DAN APLIKASI DALAM ALGORITMA SEDERHANA

Peta Konsep



Gambar 2.1 Wikipedia (2023)

N	Bilangan asli	Bilangan asli adalah angka 1, 2, 3, dll., Mungkin juga termasuk 0. Beberapa definisi, termasuk standar ISO 80000-2, memulai bilangan asli dengan 0, sesuai dengan bilangan bulat non-negatif 0, 1, 2, 3, ..., sedangkan definisi lain dimulai dengan 1.	0, 1, 2, 3, 4, 5, ... atau 1, 2, 3, 4, 5, ... N_0 dan N_1 terkadang digunakan
Z	Bilangan bulat	Bilangan bulat adalah nol (0), bilangan asli positif, atau bilangan bulat negatif dengan tanda minus. Bilangan negatif adalah	..., −5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5, ...

		penjumlahan terbalik dari bilangan positif yang sesuai. Dalam bahasa matematika, himpunan bilangan bulat sering dilambangkan dengan huruf tebal Z.	
Q	Bilangan rasional	Bilangan rasional adalah bilangan yang dapat dinyatakan sebagai hasil bagi atau pecahan dari dua bilangan bulat, pembilang p dan penyebut bukan nol q. Misalnya, ini adalah bilangan rasional, seperti bilangan bulat lainnya.	$\frac{a}{b}$ di mana a dan b adalah bilangan bulat dan b bukan 0
R	Bilangan real	Bilangan real adalah bilangan yang dapat digunakan untuk mengukur besaran satu dimensi kontinu seperti jarak, durasi, atau suhu. Di sini, kontinu berarti bahwa pasangan nilai dapat memiliki perbedaan kecil. Setiap bilangan real bisa dikatakan unik.	Batas deret konvergen bilangan rasional
C	Bilangan kompleks	Bilangan kompleks adalah elemen dari sistem bilangan yang merupakan perluasan bilangan real dengan unsur-unsur tertentu yang dilambangkan dengan i, yang disebut satuan imajiner dan memenuhi persamaan; Setiap bilangan kompleks dapat dinyatakan dalam bentuk, di mana A dan B adalah bilangan real.	a + bi di mana a dan b adalah bilangan real dan i adalah akar kuadrat formal dari −1
∴	$\mathbf{N} \subset \mathbf{Z} \subset \mathbf{Q} \subset \mathbf{R} \subset \mathbf{C}$		

Pendahuluan

Representasi angka adalah cara kita menggambarkan atau mengekspresikan nilai numerik dalam bentuk yang dapat dipahami dan digunakan oleh komputer atau manusia. Ini adalah konsep dasar dalam ilmu komputer dan matematika. Representasi angka dapat berupa desimal, biner, oktal, atau heksadesimal, tergantung pada persyaratan aplikasi dan algoritma yang digunakan (Mark & Workman, 2022). Dalam konteks algoritma sederhana, representasi angka menjadi sangat penting. Algoritma adalah langkah-langkah yang diperlukan untuk menyelesaikan tugas atau masalah. Dalam banyak kasus, algoritma memerlukan manipulasi angka, seperti penambahan, pengurangan, perkalian, atau pembagian (Hickendorff et al., 2019). Oleh karena itu, pemahaman yang baik tentang representasi angka diperlukan untuk merancang dan mengimplementasikan algoritma sederhana dengan benar.

Salah satu representasi angka yang paling umum digunakan dalam algoritma sederhana adalah representasi desimal (Maley, 2021). Angka desimal adalah angka dengan basis 10, yang berarti terdiri dari angka 0 hingga 9. Representasi desimal digunakan dalam kehidupan sehari-hari dan aplikasi seperti kalkulator. Algoritma sederhana sering mengambil angka desimal sebagai input, dan mereka dapat menghasilkan output dalam representasi desimal juga.

Selain representasi desimal, algoritma sederhana juga dapat menggunakan representasi bilangan biner. Bilangan biner adalah bilangan dengan basis 2, yang hanya terdiri dari dua bilangan yaitu 0 dan 1 (Tahir, 2021). Representasi biner sangat penting dalam dunia komputer karena komputer bekerja dengan kode biner. Banyak algoritma sederhana beroperasi pada data biner, seperti menyortir dan mencari algoritma. Representasi angka adalah bagaimana kita menggambarkan nilai numerik, dan ini sangat penting dalam algoritma sederhana. Algoritma sederhana dapat beroperasi dengan berbagai representasi angka, termasuk desimal dan biner, tergantung pada persyaratan aplikasi. Pemahaman yang baik tentang representasi angka memungkinkan perancang algoritma untuk menerapkan langkah-langkah yang benar dan menghasilkan hasil yang diinginkan.

Pembahasan

Bilangan real

Sistem bilangan desimal

Sistem bilangan desimal adalah sistem bilangan yang paling umum digunakan di dunia sehari-hari. Dalam sistem ini, kita menggunakan sepuluh angka, yaitu 0 hingga 9, untuk mewakili nilai numerik. Nilai setiap posisi angka dalam angka desimal terkait dengan pangkat 10. Misalnya, dalam angka desimal, 1234 memiliki nilai berikut:

$(1 \text{ x } 10^3) + (2 \text{ x } 10^2) + (3 \text{ x } 10^1) + (4 \text{ x } 10^0)$

Sistem bilangan desimal memiliki banyak aplikasi dalam algoritma sederhana. Salah satu aplikasinya adalah algoritma tambahan (Jang et al., 2019). Ketika kita ingin menambahkan dua angka atau lebih, kita menggunakan representasi desimal untuk angka-angka itu, dan algoritma penjumlahan sederhana akan menghitung hasil penjumlahan dalam sistem angka desimal.

Selain itu, angka desimal juga digunakan dalam algoritma penyortiran. Algoritma sederhana seperti pengurutan gelembung atau penyortiran pemilihan dapat digunakan untuk mengurutkan daftar angka desimal, naik atau turun/mengurutkan data dari kecil ke besar dan turun mengurutkan data dari besar ke kecil (Skliarova et al., 2019). Ini adalah implementasi penting karena penyortiran adalah operasi yang sering digunakan dalam banyak jenis aplikasi.

Dalam dunia pemrograman komputer, angka desimal juga digunakan dalam algoritma konversi (Rhyne, 1970). Ketika kita perlu mengonversi angka dari satu sistem angka ke sistem angka lainnya, seperti dari desimal ke biner atau sebaliknya, algoritma konversi dapat digunakan untuk melakukan operasi dengan benar.

$312{,}45 = 3 \text{ x } 10^2 + 1 \text{ x } 10^1 + 2 \text{ x } 10^0 + 4 \text{ x } 10^{-1} + 5 \text{ x } 10^{-2}$

(3 ratusan, 1 puluhan, 2 satu, 4 persepuluh, 5 per seratus)

Sistem desimal: Basis = 10, angkanya adalah (0, 1,..., 9)

Representasi standar: ± 312,45

±	3 1 2,	4 5
Tanda Angka	Bilangan bulat/bagian bulat	Bagian desimal

Representasi Nomor Float yang Dinormalisasi

Representasi bilangan float yang dinormalisasi adalah cara untuk menggambarkan bilangan real di komputer (Boehm, 2020). Dalam representasi ini, bilangan float terdiri dari tiga komponen utama: Tanda (positif atau negatif), eksponen, dan mantissa (pecahan atau bagian desimal). Representasi float yang dinormalisasi memastikan bahwa mantissa memiliki digit pertama yang selalu 1, sehingga mengoptimalkan presisi dan rentang nilai yang dapat diwakili oleh angka float.

Penerapan representasi bilangan float yang dinormalisasi dalam algoritma sederhana adalah dalam perhitungan matematis yang melibatkan bilangan real. Misalnya, algoritma sederhana seperti menghitung akar kuadrat atau membagi dua bilangan real menggunakan representasi float yang dinormalisasi. Ketika kita beroperasi dengan angka float, kita perlu memahami bagaimana mantissa dan eksponen bekerja sama untuk menghasilkan hasil yang akurat.

Representasi float yang dinormalisasi juga penting dalam pemrograman komputer karena memungkinkan kita untuk menyimpan dan memproses data yang melibatkan bilangan real dengan presisi tertentu. Misalnya, dalam pengembangan perangkat lunak untuk simulasi fisika, analisis data, kita sering menggunakan bilangan float yang dinormalisasi untuk melakukan perhitungan yang akurat.

Selain itu, representasi float yang dinormalisasi juga terkait dengan algoritma yang melibatkan perbandingan bilangan real. Dalam kondisi tertentu, kita perlu membandingkan dua bilangan float, dan memahami cara kerja representasi ini sangat penting untuk memastikan perbandingan yang benar dan hasil yang diharapkan dalam algoritma sederhana.

Representasi nomor float yang dinormalisasi:

±	$d.f_1\ f_2\ f_3\ f_4$	x	$10^{\pm n}$	, $d \neq 0$
Tanda Angka	Mantissa 4 digit angka desimal		Eksponen ±n (bilangan bulat)	

Hanya satu digit (bukan nol) yang muncul sebelum bagian desimal.

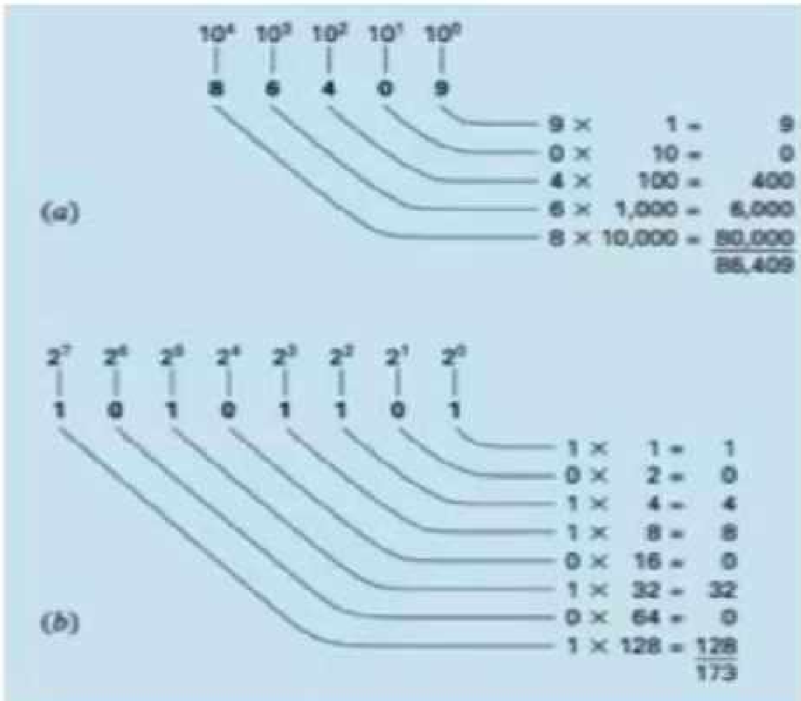
Keuntungan: Efisiensi dalam mewakili angka yang sangat kecil atau sangat besar.

Sistem Biner: Basis = 2, digit (0,1)

±	$1.\ f_1\ f_2\ f_3\ f_4$	x	$2^{\pm n}$
Tanda Angka	Mantissa		Eksponen

Contoh:

a. $(11)_2 = 1 \times 2^1 + 1 \times 2^0 = 2 + 1 = (3)_{10}$
b. $(1.101)_2 = (1 \times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3}) = 1 + 0.5 + 0 + 0{,}125 = (1.625)_{10}$



Fakta

Angka yang memiliki representasi terbatas dalam satu sistem bilangan mungkin memiliki representasi tak terbatas dalam sistem bilangan lain:

$(1{,}1)_{10} = (1{,}0001\ \ 1001\ \ 1001\ \ 100\ldots)_2$

Kita tidak akan pernah bisa mewakili 1.1 secara terbatas dalam sistem biner.

Karena untuk dapat mewakilinya secara terbatas, satu-satunya faktor prima dalam penyebut harus 2

Contoh: 0,1 atau $\frac{1}{10}$ faktor prima dalam penyebutnya adalah 2 dan 5.

Tabel 2.1 Pecahan, desimal, dan biner

Fraction	Decimal	Binary	Fractional approximation
1/1	1 or 0.999...	1 or 0.111...	1/2 + 1/4 + 1/8...
1/2	0.5 or 0.4999...	0.1 or 0.0111...	1/4 + 1/8 + 1/16 . . .
1/3	0.333...	0.010101...	1/4 + 1/16 + 1/64 . . .
1/4	0.25 or 0.24999...	0.01 or 0.00111...	1/8 + 1/16 + 1/32 . . .
1/5	0.2 or 0.1999...	0.00110011...	1/8 + 1/16 + 1/128 . . .
1/6	0.1666...	0.0010101...	1/8 + 1/32 + 1/128 . . .
1/7	0.142857142857...	0.001001...	1/8 + 1/64 + 1/512 . . .
1/8	0.125 or 0.124999...	0.001 or 0.000111...	1/16 + 1/32 + 1/64 . . .
1/9	0.111...	0.000111000111...	1/16 + 1/32 + 1/64 . . .
1/10	0.1 or 0.0999...	0.000110011...	1/16 + 1/32 + 1/256 . . .
1/11	0.090909...	0.00010111010001011101...	1/16 + 1/64 + 1/128 . . .
1/12	0.08333...	0.00010101...	1/16 + 1/64 + 1/256 . . .
1/13	0.076923076923...	0.000100111011000100111011...	1/16 + 1/128 + 1/256 . . .
1/14	0.0714285714285...	0.0001001001...	1/16 + 1/128 + 1/1024 . . .
1/15	0.0666...	0.00010001...	1/16 + 1/256 . . .
1/16	0.0625 or 0.0624999...	0.0001 or 0.0000111...	1/32 + 1/64 + 1/128 . . .

Floating-Point Standard IEEE 754

IEEE 754 Floating-Point Standard adalah standar yang digunakan untuk mewakili bilangan real dalam format biner di komputer. Standar ini menentukan bagaimana komputer menyimpan, memproses, dan melakukan operasi matematika pada bilangan real. IEEE 754 menjelaskan bilangan float dan ganda dalam tiga bagian utama: tanda (positif atau negatif), eksponen (kekuatan dasar), dan mantissa (bagian pecahan atau desimal). Representasi ini memungkinkan komputer untuk mengatasi keterbatasan dalam mewakili bilangan real dengan tepat.

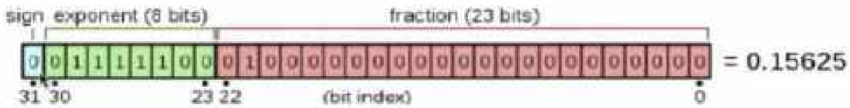
Penerapan IEEE 754 Floating-Point Standard dalam algoritma sederhana sangat luas. Salah satu aplikasinya adalah dalam perhitungan aritmatika dasar seperti penjumlahan, pengurangan, perkalian dan pembagian bilangan real. Algoritma sederhana yang melakukan operasi ini menggunakan representasi IEEE 754 untuk menghasilkan hasil yang mendekati nilai sebenarnya.

Selain itu, algoritma numerik seperti perhitungan akar kuadrat, trigonometri, atau logaritma juga menggunakan representasi IEEE 754. Dalam konteks ini, algoritma sederhana memproses input dalam format IEEE 754, melakukan perhitungan matematis, dan mengembalikan hasil dalam format yang sama. Pemahaman yang baik tentang representasi IEEE 754 penting dalam memastikan keakuratan perhitungan numerik dalam aplikasi ini.

Selain operasi matematika, representasi IEEE 754 juga digunakan dalam algoritma yang melibatkan perbandingan bilangan real. Dalam kondisi tertentu, seperti menyortir data numerik atau memilih elemen terbesar/terkecil dalam daftar bilangan real, representasi IEEE 754 digunakan untuk membandingkan nilai numerik dengan benar.

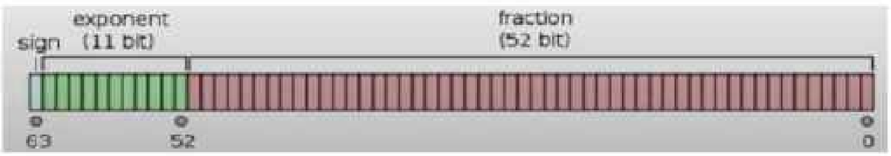
Presisi tunggal (representasi 32-bit)

Tanda 1-bit + Eksponen 8-bit + Pecahan 23-bit



Presisi ganda (representasi 64-bit)

Tanda 1-bit + Eksponen 11-bit + Pecahan 52-bit



Panduan Standar Sistem Biner

Tabel 2.2 Standar sistem biner

Name	Common name	Base	Significand bits or digits	Decimal digits	Exponent bits	Decimal E max	Exponent bias	E min	E max	Notes
binary16	Half precision	2	11	3.31	5	4.51	$2^4-1 = 15$	−14	+15	not basic
binary32	Single precision	2	24	7.22	8	38.23	$2^7-1 = 127$	−126	+127	
binary64	Double precision	2	53	15.95	11	307.95	$2^{10}-1 = 1023$	−1022	+1023	
binary128	Quadruple precision	2	113	34.02	15	4931.77	$2^{14}-1 = 16383$	−16382	+16383	
binary256	Octuple precision	2	237	71.34	19	78913.2	$2^{18}-1 = 262143$	−262142	+262143	not basic
decimal32		10	7	7	7.58	96	101	−95	+96	not basic
decimal64		10	16	16	9.58	384	398	−383	+384	
decimal128		10	34	34	13.58	6144	6176	−6143	+6144	

$$\text{Single} \longrightarrow \text{value} = (-1)^{\text{sign}} \times 2^{(e-127)} \times \left(1 + \sum_{i=1}^{23} b_{23-i} 2^{-i}\right)$$

$$\text{Double} \longrightarrow (-1)^{\text{sign}} \left(1 + \sum_{i=1}^{52} b_{52-i} 2^{-i}\right) \times 2^{e-1023}$$

Angka Penting (Significant Figures)

Angka signifikan adalah jumlah digit yang diperhitungkan dalam kuantitas yang diukur atau dihitung. Angka signifikan adalah digit dalam representasi angka yang memberikan informasi penting tentang tingkat presisi atau akurasi angka tersebut. Dalam representasi bilangan real dalam format seperti bilangan floating-point atau eksponensial, angka-angka ini menunjukkan seberapa rinci informasi disimpan dalam angka tersebut. Dalam konteks algoritma sederhana, angka signifikan memiliki peran penting dalam menentukan akurasi hasil perhitungan.

Penerapan angka-angka penting dalam algoritma sederhana dapat ditemukan dalam perhitungan aritmatika dasar. Ketika dua atau lebih bilangan real dengan angka signifikan yang berbeda digunakan dalam perhitungan seperti penjumlahan atau perkalian, keakuratan hasil perhitungan dapat dipengaruhi oleh angka signifikan terkecil dari semua angka yang terlibat. Algoritma sederhana ini akan mempertahankan sejumlah digit signifikan sehingga hasil perhitungan sesuai dengan tingkat presisi yang diinginkan.

Selain itu, dalam algoritma yang melibatkan pembulatan, seperti pembulatan bilangan real ke atas atau ke bawah ke bilangan bulat terdekat, angka signifikan juga memainkan peran penting. Algoritma ini akan mempertimbangkan angka-angka penting untuk menentukan hasil pembulatan yang sesuai. Misalnya, jika kita ingin membulatkan bilangan real dengan 3 digit signifikan ke bilangan bulat terdekat, kita akan memperhatikan tiga digit terakhir dalam bilangan tersebut.

Angka signifikan juga memiliki peran dalam algoritma yang melibatkan perbandingan bilangan real. Sebagai perbandingan, algoritma sederhana membandingkan digit signifikan untuk menentukan hubungan antara dua angka, seperti mana yang lebih besar, kurang, atau sama.

Contoh 1-Kalkulator:

Misalkan kita ingin menghitung:

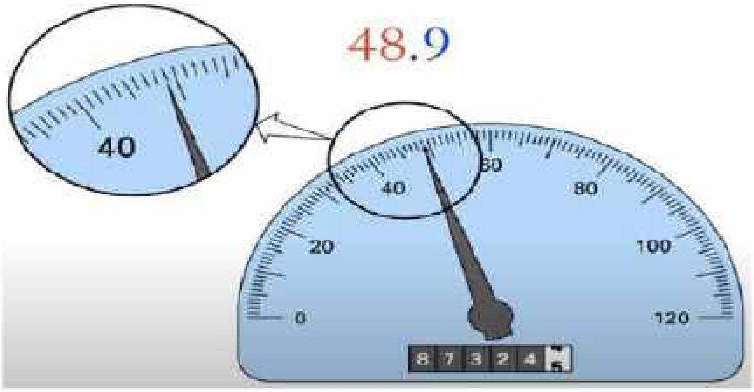
3,57 x 2,13 dengan menggunakan kalkulator dengan 2 digit setelah koma:

3,57 x 2,13 = 7,60

Jawaban yang benar: menggunakan Ms. Excel kemudian diperoleh yaitu:

3.57	2.13	7.604100

Contoh-2:



Gambar 2.2 Alat pengukur berat badan

Akurasi dan Presisi

Akurasi berkaitan dengan seberapa dekat nilai perkiraan dengan nilai sebenarnya. Akurasi dan presisi adalah dua konsep penting dalam algoritma sederhana yang berhubungan dengan sejauh mana hasil algoritma mendekati nilai yang diharapkan atau dianggap benar. Meskipun sering digunakan secara bergantian, mereka memiliki arti yang sedikit berbeda.

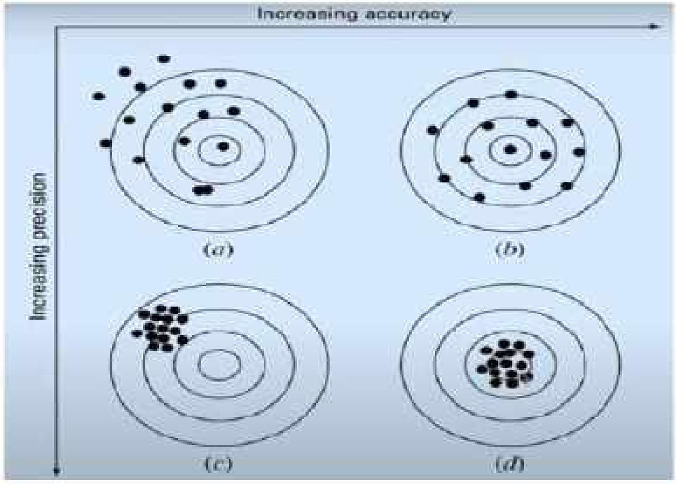
Akurasi mengacu pada seberapa dekat hasil algoritma dengan nilai sebenarnya atau solusi yang tepat. Ini adalah ukuran sejauh mana algoritma mampu menghasilkan hasil yang tepat sesuai dengan tingkat presisi yang diinginkan. Dalam algoritma sederhana, seperti perhitungan matematis dasar, akurasi diukur dengan membandingkan hasil algoritma dengan nilai referensi atau solusi yang diketahui. Semakin dekat ke nilai referensi, semakin tinggi tingkat akurasi algoritma.

Presisi, disisi lain adalah ukuran sejauh mana hasil algoritma dekat satu sama lain ketika algoritma diterapkan beberapa kali dengan input yang sama atau serupa. Ini mengacu pada konsistensi hasil algoritma di beberapa iterasi atau uji coba. Algoritma yang memiliki tingkat presisi tinggi akan menghasilkan hasil yang serupa bila dijalankan berulang kali dengan input yang sama.

Aplikasi akurasi dan presisi dalam algoritma sederhana dapat ditemukan dalam berbagai konteks. Dalam perhitungan matematis dasar seperti penjumlahan atau perkalian, akurasi sangat penting untuk menghasilkan hasil yang mendekati nilai sebenarnya. Di sisi lain, dalam algoritma yang melakukan iterasi seperti pencarian biner, presisi adalah fokus karena kami ingin mendapatkan hasil yang konsisten dan akurat dalam setiap iterasi.

Selain itu, algoritma yang melibatkan data pengukuran atau sensor juga harus memperhatikan akurasi dan presisi. Dalam ilmu data dan statistik, presisi mengacu pada sejauh mana hasil pengukuran atau perkiraan data berdekatan satu sama lain di berbagai pengukuran atau analisis. Akurasi di sini mengacu pada sejauh mana hasil analisis mendekati nilai sebenarnya dari data.

Presisi/akurasi berkaitan dengan seberapa dekat hasil estimasi dengan hasil estimasi sebelumnya.



Gambar 2.3. Ilustrasi presisi/akurasi

Pembulatan dan Pemotongan

Pembulatan dan pemotongan adalah dua teknik yang digunakan dalam algoritma untuk mengubah hasil perhitungan atau data numerik menjadi bentuk yang lebih sesuai dengan kebutuhan atau presisi yang diinginkan.

Pembulatan adalah teknik yang digunakan untuk membulatkan angka ke bilangan bulat terdekat atau sejumlah digit setelah koma desimal. Misalnya, jika kita memiliki hasil perhitungan 3,6 pembulatan ke bilangan bulat terdekat akan menghasilkan 4. Teknik ini penting dalam algoritma sederhana seperti ketika kita ingin mengubah hasil perhitungan kontinu menjadi bilangan bulat yang lebih mudah dipahami manusia.

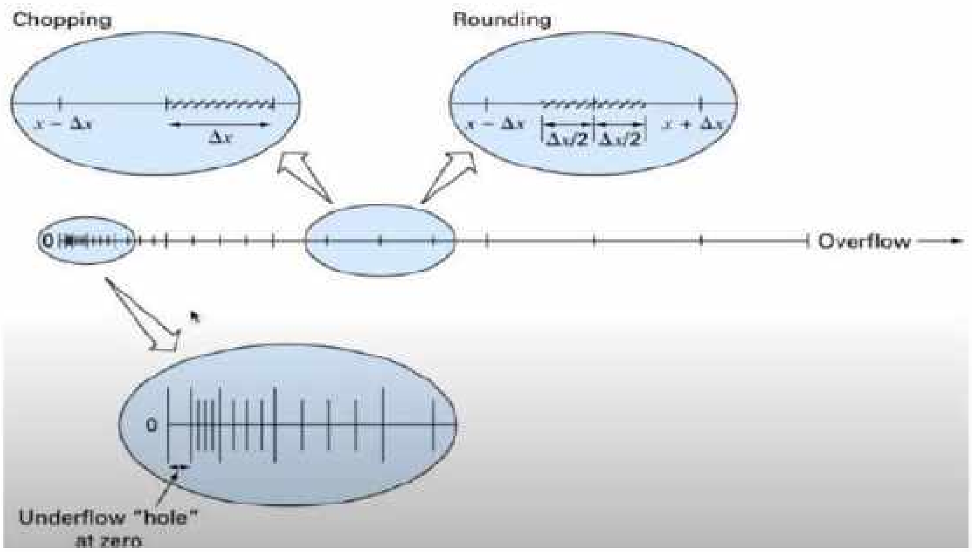
Memotong, di sisi lain, adalah teknik yang digunakan untuk menghapus sejumlah digit setelah koma desimal tanpa membulatkannya. Misalnya, jika kita memiliki hasil perhitungan 3,678 pemotongan akan menghasilkan 3. Ini sering digunakan dalam algoritma sederhana ketika kita ingin membuang angka yang kurang signifikan atau tidak diperlukan dalam hasil akhir.

Implementasi pembulatan dan pemotongan dalam algoritma sederhana sangat umum. Dalam perhitungan keuangan, seperti menghitung pajak atau pembayaran, pembulatan sering digunakan untuk memastikan hasil akhir memiliki presisi yang sesuai untuk mata uang yang digunakan. Di sisi lain, pemotongan digunakan dalam konteks di mana jumlah digit desimal yang lebih rendah tidak relevan atau tidak diinginkan, seperti dalam presentasi grafis atau laporan yang memerlukan tampilan yang lebih bersih.

Selain itu, dalam algoritma yang melibatkan pemrosesan data numerik atau statistik, pembulatan dan pemotongan dapat digunakan untuk mengontrol tingkat presisi hasil. Ini memungkinkan kami untuk mengontrol jumlah digit desimal yang ditampilkan atau digunakan dalam analisis data, sesuai dengan kebutuhan spesifik tugas.

Pembulatan: Ganti angka dengan angka terdekat

Memotong: Buang semua digit yang tersisa/ekstra



Gambar 2.4 Pembulatan dan pemotongan

Definisi Error-The Actual Error

Kesalahan sebenarnya adalah perbedaan antara hasil yang dihasilkan oleh suatu algoritma atau percobaan dan nilai yang dianggap sebagai "nilai sebenarnya" atau solusi yang benar. Kesalahan dapat timbul dari berbagai sumber, seperti kesalahan pengukuran, pembulatan, aproksimasi, atau kesalahan dalam algoritma itu sendiri (Ralston & Rabinowitz, 2001). Dalam analisis numerik dan ilmu lainnya, memahami kesalahan dan kemampuan untuk mengukurnya adalah aspek penting dalam memastikan akurasi dan keandalan hasil.

Kesalahan sejati adalah ukuran sejauh mana hasil yang diberikan oleh suatu algoritma atau eksperimen mendekati nilai yang dianggap benar atau solusi yang benar. Kesalahan sebenarnya adalah perbedaan antara hasil algoritma dan nilai yang benar atau solusi yang benar, yang seringkali sulit atau bahkan tidak mungkin untuk diketahui dengan pasti. Oleh karena itu, kesalahan sebenarnya adalah perkiraan sejauh mana hasil kami dapat diandalkan dan sesuai dengan tujuan yang diinginkan.

Aplikasi konsep kesalahan dan kesalahan aktual dalam algoritma sederhana dapat ditemukan dalam berbagai konteks. Dalam perhitungan numerik, kesalahan sering diukur untuk mengukur keakuratan hasil perhitungan. Misalnya, dalam algoritma sederhana untuk menghitung akar kuadrat, kesalahan dapat dihitung dengan membandingkan hasil algoritma dengan nilai akar kuadrat yang diketahui. Semakin kecil kesalahan, semakin dekat hasil algoritma ke nilai yang benar.

Selain itu, dalam ilmu data, konsep kesalahan digunakan untuk mengevaluasi model statistik. Model-model ini menghasilkan prediksi atau perkiraan, dan kesalahan digunakan untuk mengukur sejauh mana prediksi mendekati nilai aktual. Dalam hal ini, memahami kesalahan membantu kita memahami kualitas model dan mengidentifikasi area di mana model perlu ditingkatkan.

Kesalahan dapat dihitung jika nilai sebenarnya dapat dihitung:

Kesalahan Absolut = $E_t = |True\ value - Approximate\ value|$

Kesalahan Relatif (dalam persen)

$$\varepsilon_t = \left|\frac{True\ value - Approximate\ value}{True\ value}\right| 100\%$$

Definisi Error–Estimated Error

Estimasi kesalahan adalah upaya untuk mengukur atau memperkirakan besarnya kesalahan dengan tujuan untuk memahami sejauh mana hasil yang diperoleh dapat diandalkan.

Penerapan konsep kesalahan estimasi dalam algoritma sederhana dapat ditemukan dalam berbagai situasi. Misalnya, ketika kita melakukan perhitungan matematika sederhana seperti pembagian dengan angka yang memiliki banyak digit desimal, seringkali kita harus membulatkan hasil perhitungan agar lebih mudah dipahami. Dalam hal ini, kita tahu bahwa ada kesalahan karena pembulatan, dan kita dapat memperkirakan seberapa besar kesalahan tersebut.

Dalam pengolahan data dan statistik, mengukur kesalahan estimasi penting dalam mengukur akurasi estimasi parameter statistik. Misalnya, jika kita ingin memperkirakan rata-rata populasi berdasarkan sampel, kita akan menghitung rata-rata sampel dan juga mengukur seberapa besar kesalahan standar perkiraan. Perkiraan kesalahan dalam kasus ini membantu kami memahami sejauh mana perkiraan kami dapat dipercaya sebagai representasi dari populasi sebenarnya.

Selain itu, dalam pemodelan numerik seperti mendekati fungsi matematika kompleks dengan polinomial sederhana, kita sering menggunakan kesalahan yang diperkirakan untuk memahami sejauh mana perkiraan kita mendekati fungsi sebenarnya. Ini membantu kami mengevaluasi seberapa baik model kami mewakili fenomena yang ingin kami akses.

Perkiraan kesalahan digunakan ketika nilai sebenarnya tidak diketahui:

Estimasi Kesalahan Absolut:

$$E_a = |Present\ value - Previous\ value|$$

Estimasi Kesalahan Relatif (dalam persen):

$$\varepsilon_t = \left|\frac{Present\ value - Previous\ value}{Present\ value}\right| 100\%$$

Notasi

Hasil estimasi dikatakan benar hingga n digit desimal jika:

$$|Error| \leq 10^{-n}$$

Hasil estimasi dikatakan benar hingga n digit desimal pembulatan jika:

$$|Error| \leq \frac{1}{2} x\ 10^{-n}$$

Kesimpulan

Representasi angka adalah cara kita menggambarkan nilai numerik dalam bentuk yang dapat dipahami dan digunakan oleh komputer atau manusia. Ada berbagai sistem representasi angka, termasuk desimal, biner, oktal, dan heksadesimal yang digunakan dalam berbagai aplikasi dan algoritma. Penerapan representasi angka dalam algoritma sederhana sangat penting. Algoritma adalah serangkaian langkah yang diperlukan untuk menyelesaikan tugas atau masalah. Dalam banyak kasus, algoritma memerlukan manipulasi angka, seperti penambahan, pengurangan, perkalian, atau pembagian. Oleh karena itu, pemahaman yang baik tentang representasi angka diperlukan untuk merancang dan mengimplementasikan algoritma sederhana dengan benar.

Representasi angka desimal adalah yang paling umum digunakan dalam kehidupan sehari-hari dan dalam aplikasi seperti kalkulator. Banyak algoritma sederhana mengambil angka desimal sebagai input dan menghasilkan output dalam representasi desimal juga. Selain representasi desimal, algoritma sederhana juga dapat menggunakan representasi bilangan biner. Representasi biner sangat penting dalam dunia komputer karena komputer bekerja dengan kode biner. Banyak algoritma sederhana beroperasi pada data biner, seperti menyortir dan mencari algoritma

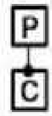
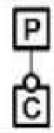
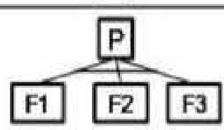
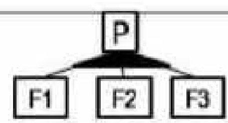
Angka yang memiliki representasi terbatas dalam satu sistem bilangan mungkin memiliki representasi tak terbatas dalam sistem bilangan lain. Dalam representasi floating-point yang dinormalisasi, bilangan real diwakili oleh tanda, eksponen, dan mantissa, di mana mantissa selalu memiliki digit pertama sama dengan 1. Hal ini memungkinkan representasi bilangan real yang efisien dan presisi tinggi. Penerapannya dalam algoritma sederhana terutama berfokus pada perhitungan matematis yang melibatkan bilangan real, seperti penambahan, pengurangan, perkalian dan pembagian. Representasi ini membantu algoritma sederhana dalam menghasilkan hasil yang mendekati nilai sebenarnya dengan tingkat presisi yang tinggi.

Dengan memahami representasi floating-point yang dinormalisasi, perancang algoritma dapat mengoptimalkan perhitungan numerik dalam berbagai aplikasi. Meskipun representasi ini mungkin lebih kompleks daripada bilangan bulat, penggunaannya dapat meningkatkan akurasi dan presisi dalam algoritma sederhana, memastikan bahwa hasil perhitungan mendekati nilai sebenarnya dengan tingkat kesalahan minimal. Pembulatan dan pemotongan membuatnya efisien untuk mewakili jumlah yang sangat kecil atau sangat besar. Representasi kesalahan tergantung pada bit yang digunakan.

3 KONSEP DASAR PREPOSISI DAN PREPOSISI DALAM ALGORITMA PEMROGRAMAN

Peta Konsep

Tabel 3.1 Peta konsep logika proposisional (Lopez-Herrejon & Egyed, 2010)

Name	Diagram Notation	Propositional Logic
Mandatory		$P \Leftrightarrow C$
Optional		$C \Rightarrow P$
Exclusive-Or		$(F1 \Leftrightarrow (\neg F2 \wedge \neg F3 \wedge P)) \wedge$ $(F2 \Leftrightarrow (\neg F1 \wedge \neg F3 \wedge P)) \wedge$ $(F3 \Leftrightarrow (\neg F1 \wedge \neg F2 \wedge P))$
Inclusive-Or		$P \Leftrightarrow F1 \vee F2 \vee F3$
Requires	Cross feature arrow	$A \Rightarrow B$
Excludes	Cross feature arrow	$A \Rightarrow \neg B \equiv \neg(A \wedge B)$

Pendahuluan

Logika proposisional dan konsep preposisi memainkan peran penting dalam bidang algoritma pemrograman, menjembatani kesenjangan antara dunia abstrak logika matematika dan ranah praktis ilmu komputer (Sack, 2019). Kedua subjek yang saling berhubungan ini memberikan kerangka dasar untuk penalaran tentang kebenaran, nilai-nilai pernyataan, dan hubungan antara berbagai elemen dalam algoritma. Dalam pengantar ini, kita akan mengeksplorasi pentingnya logika proposisional dan bagaimana preposisi menemukan aplikasi dalam algoritma pemrograman.

Logika Proposisional

Logika proposisional, sering disebut sebagai logika sentential atau kalkulus proposisional, berfungsi sebagai tulang punggung logis ilmu komputer dan membentuk dasar untuk penalaran logis dalam algoritma (Chowdhary, 2020). Pada intinya, logika proposisional berkaitan dengan proposisi, yang merupakan pernyataan yang bisa benar atau salah. Ini menawarkan satu set penghubung logis, seperti AND, OR, dan NOT, yang memungkinkan programmer untuk memanipulasi proposisi, memungkinkan mereka untuk membuat keputusan berdasarkan informasi dalam algoritma. Dengan menggunakan tabel kebenaran dan aturan formal inferensi, logika proposisional memastikan bahwa programmer dapat bernalar secara logis, yang sangat penting untuk desain algoritma, kebenaran, dan optimasi.

Preposisi sebagai Konstruksi Fundamental

Dalam konteks algoritma pemrograman, preposisi bukanlah konektor linguistik yang ditemukan dalam bahasa alami melainkan konstruksi mendasar yang digunakan untuk mengekspresikan hubungan dan kondisi antara elemen program yang berbeda (Chowdhary & Chowdhary, 2020). Preposisi membantu programmer menentukan aturan dan batasan yang memandu aliran suatu algoritma. Misalnya, "jika" "sementara" dan "sampai" adalah preposisi yang digunakan untuk menentukan kapan tindakan spesifik harus diambil berdasarkan kondisi tertentu. Preposisi ini menentukan logika aliran kontrol dalam algoritma, memastikan bahwa eksekusi kode terjadi dengan cara yang terkontrol dan bermakna.

Pengambilan Keputusan yang Logis

Integrasi logika proposisional dan preposisi memungkinkan programmer untuk membuat keputusan yang tepat dan tepat dalam algoritma mereka (Baden et al., 2022). Mereka dapat menggunakan penghubung logis untuk menggabungkan proposisi dan mengekspresikan kondisi kompleks yang menentukan bagaimana suatu algoritma berperilaku. Pernyataan kondisional, loop, dan struktur percabangan dalam pemrograman sangat bergantung pada konstruksi logis ini. Misalnya, algoritma mungkin menyertakan pernyataan kondisional yang mengeksekusi blok kode tertentu hanya jika proposisi tertentu berlaku. Pengambilan keputusan logis ini adalah inti dari penulisan algoritma yang efisien, efektif, dan andal.

Penanganan dan Validasi Kesalahan

Preposisi dan logika proposisional juga sangat berharga untuk penanganan kesalahan dan validasi data dalam algoritma (Bordel et al., 2021). Dengan merumuskan preposisi yang memeriksa kondisi kesalahan tertentu atau memvalidasi data input, programmer dapat memastikan integritas dan keandalan algoritma mereka. Misalnya, algoritma yang memproses input pengguna dapat menggunakan logika proposisional untuk memverifikasi apakah input mematuhi kriteria tertentu, mencegah kesalahan yang tidak diinginkan atau kerentanan keamanan.

Pembahasan

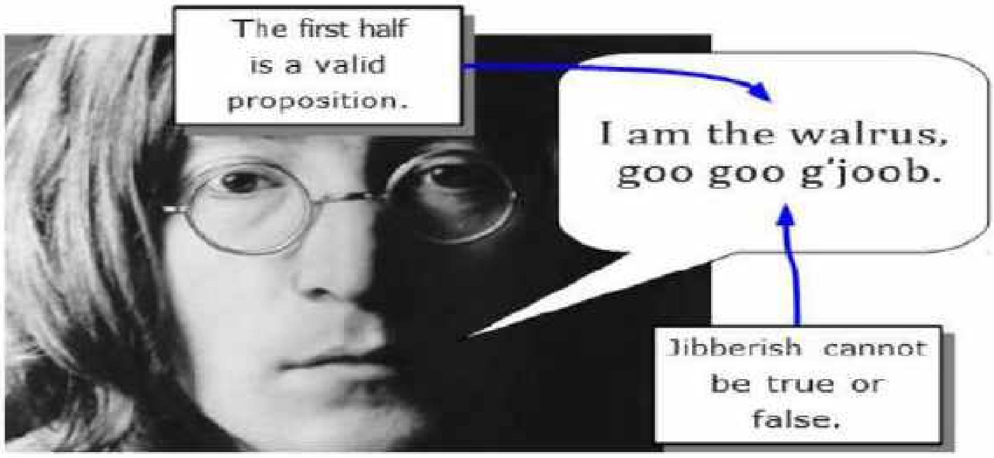
Logika Proposisional

Proposisi adalah pernyataan yang, dengan sendirinya, benar atau salah (Dewey, 1941).

- Anak anjing lebih manis dari anak kucing.
- Anak kucing lebih manis dari anak anjing.
- Bolt dapat berlari lebih cepat dari semua orang di ruangan ini.
- CS103 berguna untuk pesta koktail.
- Ini adalah entri terakhir dalam daftar ini.
- Saya seorang wanita lajang.
- Tempat ini akan meledak.
- Pesta besar ada di rumah malam ini.
- Kita bisa menari jika kita mau.
- Kita bisa meninggalkan teman-temanmu.

Hal-hal yang Bukan Proposisi

Hal-hal yang Bukan Proposisi



- Logika proposisional adalah sistem matematika untuk penalaran tentang proposisi dan bagaimana mereka berhubungan satu sama lain.
- Logika proposisional memungkinkan kita secara formal menyandikan bagaimana kebenaran berbagai proposisi mempengaruhi kebenaran proposisi lain.
- Tentukan apakah kombinasi proposisi tertentu selalu, kadang-kadang, atau tidak pernah benar.
- Tentukan apakah kombinasi proposisi tertentu secara logis memerlukan kombinasi lain.

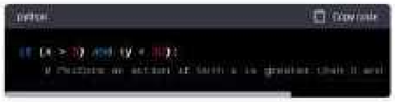
Variabel dan Penghubung

- Logika proposisional adalah sistem matematika formal yang sintaksisnya ditentukan secara kaku (Braüner & Ghilardi, 2007).
- Setiap pernyataan dalam logika proposisional terdiri dari variabel proposisional yang digabungkan melalui penghubung logis (Allan, 2023).
- Setiap variabel mewakili beberapa proposisi, seperti "Anda menginginkannya" atau "Anda seharusnya memasang cincin di atasnya."
- Penghubung menyandikan bagaimana proposisi terkait, seperti "Jika Anda menginginkannya, Anda seharusnya memasang cincin di atasnya." (Bloom et al., 1980).

Variabel Proposisional

- Setiap proposisi akan diwakili oleh variabel proposisional.
- Variabel proposisional biasanya direpresentasikan sebagai huruf kecil, seperti p, q, r, s, dll.
- Jika kita membutuhkan lebih banyak, kita dapat menggunakan subskrip: p1, p2, dll.
- Setiap variabel dapat mengambil salah satu dari dua nilai: true atau false.

Pernyataan Bersyarat:

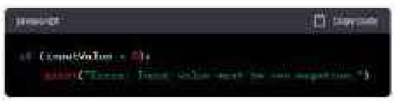


Dalam contoh ini, operator dan digunakan untuk menggabungkan dua preposisi. Blok kode akan mengeksekusi hanya jika kedua preposisi (x > 5) dan (y < 10) benar.

Kontrol Loop:

Di sini, preposisi 'while' mengontrol loop, memastikan bahwa ia terus mengeksekusi hingga kondisi (userInput != "exit") menjadi false.

Penanganan Kesalahan:



Dalam hal ini, preposisi (inputValue < 0) digunakan untuk memvalidasi data input. Jika kondisi ini benar, pesan galat akan ditampilkan.

Percabangan logis

Contoh ini menunjukkan bagaimana preposisi digunakan untuk membuat cabang logis. Tergantung pada nilai variabel "suhu", blok kode yang berbeda akan dijalankan.

Fungsi Boolean

Dalam fungsi ini, preposisi '(angka% 2 == 0)' menentukan apakah angka yang diberikan genap. Fungsi mengembalikan nilai Boolean, dan pernyataan 'if' menggunakan hasilnya untuk membuat keputusan.

Contoh-contoh ini menunjukkan bagaimana logika proposisional dan preposisi sangat penting untuk algoritma pemrograman. Mereka memungkinkan programmer untuk menciptakan kondisi, loop, dan struktur pengambilan keputusan yang mengontrol aliran dan perilaku algoritma, memastikan bahwa kode dijalankan secara logis dan akurat sebagai respons terhadap input dan kondisi yang berbeda.

(a) FOL berkaitan dengan penalaran deduktif yang mengaktifkan penggunaan 'penghubung proposisional' seperti dan, atau, jika, tidak, dan pada penggunaan 'quantiers' seperti setiap, beberapa, tidak. Tetapi dalam bahasa biasa (termasuk bahasa biasa matematika informal) operator logis ini bekerja dengan cara yang sangat kompleks, memperkenalkan jenis ketidakjelasan dan kemungkinan ambiguitas yang tentu ingin kita hindari dalam argumen yang transparan secara logis. Apa yang harus dilakukan sejak zaman Aristoteles, para ahli logika telah menggunakan strategi 'membagi dan menaklukkan' yang melibatkan pengenalan bahasa yang disederhanakan dan disiplin ketat. Bagi Aristoteles, bahasanya yang teratur adalah sebuah fragmen dari bahasa Yunani yang sangat kaku; Bagi kami, bahasa kami yang teratur sepenuhnya merupakan konstruksi formal artisial. Tapi bagaimanapun juga, rencananya adalah kita mengatasi bentangan penalaran dengan merumuskan ulang dalam bahasa yang sesuai dengan operator logis yang jauh lebih rapi, dan kemudian kita dapat mengevaluasi penalaran setelah disusun kembali ke dalam bentuk yang berperilaku lebih baik ini. Dengan cara ini, kita memiliki pembagian kerja. Pertama, kami mengklarifikasi struktur yang dimaksudkan dari argumen asli dengan menerjemahkannya ke dalam bahasa yang disederhanakan / diformalkan yang tidak ambigu. Kedua, ada urusan terpisah untuk menilai validitas argumen resimen yang dihasilkan.

Kami akan menggunakan bahasa formal yang sesuai yang berisi ,khususnya, pengganti disiplin rapi untuk penghubung proposisional dan, atau, jika, tidak (dilambangkan secara standar ∧ ∨ → , ~) ditambah pengganti untuk quantifier bahasa biasa (kira-kira, menggunakan x untuk setiap x sedemikian rupa sehingga . . . , dan y untuk beberapa y sedemikian rupa sehingga . . .).∀ ∃

Meskipun kesenangan benar-benar dimulai setelah kita memiliki quantifier dalam permainan, sangat membantu untuk mengembangkan FOL dalam dua tahap utama:

(I) Kita mulai dengan memperkenalkan bahasa yang aparat logis bawaannya hanya terdiri dari penghubung proposisional dan kemudian mendiskusikan logika proposisional argumen yang dibingkai dalam bahasa-bahasa ini. Ini memberi kita pengaturan yang sangat mudah dikelola untuk pertama kali menemukan berbagai macam konsep dan strategi logis.

(II) Kami kemudian melanjutkan untuk mengembangkan sintaksis dan semantik bahasa formal yang lebih kaya yang menambahkan aparat kuantisasi orde pertama dan mengeksplorasi logika argumen yang diterjemahkan ke dalam bahasa tersebut. Jadi, mari kita sedikit lebih detail tentang tahap (I) di bagian ini, dan kemudian kita akan beralih ke tahap (II) di bagian berikutnya.

(b) Pertama-tama kita melihat, kemudian, pada sintaksis bahasa proposisional, mendefinisikan apa yang dianggap sebagai rumus yang terbentuk dengan baik (wff) dari bahasa-bahasa tersebut. Kita mulai dengan pasokan wff 'atomik' proposisional, karena mungkin P, Q, R, dan pasokan operator logis, biasanya, ∧ ∨ → ~ ditambah mungkin konstanta absurditas yang selalu salah ⊥. Kami kemudian memiliki aturan untuk membangun wff 'molekuler', seperti jika A dan B adalah wff, begitu juga (A → B).

Jika Anda telah menemukan bahasa semacam ini, Anda sekarang perlu mengetahui bagaimana membuktikan berbagai hal tentang mereka yang tampak jelas dan bahwa Anda mungkin sebelumnya menerima begitu saja misalnya, bahwa 'bracketing bekerja' untuk memblokir ambiguitas seperti $P \vee Q \wedge R$ sehingga setiap rumus yang terbentuk dengan baik memiliki penguraian unik yang tidak ambigu.

(c) Di sisi semantik, kita membutuhkan gagasan penilaian untuk bahasa proposisional. Kita mulai dengan penugasan nilai-kebenaran, benar vs salah, ke rumus atom, blok bangunan dasar bahasa kita. Kami sekarang menarik interpretasi 'kebenaran-fungsional' dari penghubung: kami memiliki aturan seperti ($A \rightarrow B$) benar jika dan hanya jika A salah atau B benar atau keduanya yang menentukan nilai-kebenaran kompleks adalah fungsi dari nilai-kebenaran konstituennya. Dan aturan-aturan ini kemudian x bahwa setiap wff - betapapun rumitnya - ditentukan untuk menjadi pasti benar atau pasti salah (satu atau yang lain, tetapi tidak keduanya) pada penilaian tertentu dari komponen atomnya. Asumsi inti ini khas dari semantik dua nilai klasik.

(d) Bahkan pada titik awal ini, pertanyaan muncul. Misalnya, seberapa memuaskan representasi kondisional informal jika P maka Q dengan rumus $P \rightarrow Q$ yang menggunakan penghubung panah kebenaran-fungsional? Dan mengapa membatasi diri kita hanya pada segelintir kecil penghubung yang berfungsi dengan kebenaran? Anda tidak ingin terlalu terjerat dengan pertanyaan pertama, meskipun Anda perlu mencari tahu mengapa kami mewakili kondisional dalam FOL dengan cara yang kami lakukan. Adapun pertanyaan kedua, ini adalah teorema awal bahwa setiap fungsi kebenaran sebenarnya dapat diekspresikan hanya dengan menggunakan segelintir penghubung.

(e) Sekarang sepasang definisi penting (kita mulai menggunakan 'iff' sebagai singkatan standar untuk 'jika dan hanya jika'):

Sebuah wff A dari bahasa proposisional adalah tautologi iff itu benar pada setiap penugasan nilai ke atom yang relevan.

Satu set wff Γ secara tautologis memerlukan A iff setiap penugasan nilai ke atom yang relevan yang membuat semua kalimat dalam Γ benar membuat A benar juga.

Jadi gagasan tentang entailment tautologis bertujuan untuk mengatur gagasan tentang argumen yang valid secara logis berdasarkan cara penghubung muncul dalam premisses dan kesimpulannya.

Anda perlu menjelajahi beberapa sifat utama dari hubungan entailment semantik ini. Dan perhatikan bahwa dalam kasus yang agak khusus ini, kita dapat secara mekanis menentukan apakah Γ memerlukan A, misalnya dengan 'tes tabel kebenaran' (setidaknya selama hanya ada banyak w dalam Γ, dan karenanya hanya terbatas banyak atom yang relevan untuk dikhawatirkan).

(f) Presentasi buku teks yang berbeda mengisi langkah-langkah (b) hingga (e) dapat ke tingkat detail yang berbeda, tetapi cerita dasarnya tetap sama. Namun, sekarang jalan bercabang. Untuk topik berikutnya yang biasa akan menjadi sistem deduktif formal di mana kita dapat membangun derivasi langkah-demi-langkah kesimpulan dari kesalahan dalam logika proposisional. Ada berbagai sistem seperti itu untuk dipilih, dan saya akan menyebutkan tidak kurang dari S3.

Sistem pembuktian yang berbeda untuk logika proposisional klasik akan (seperti yang Anda harapkan) setara - yang berarti bahwa, mengingat beberapa premisses, kita dapat memperoleh kesimpulan yang sama di setiap sistem. Namun, sistem ini sangat berbeda dalam daya tarik intuitif dan keramahan penggunanya, serta dalam beberapa fitur yang lebih teknis. Namun, perhatikan: selain melihat beberapa contoh ilustratif, kami tidak akan terlalu tertarik untuk menghasilkan banyak derivasi di dalam sistem bukti yang dipilih; Fokusnya adalah pada penetapan hasil tentang sistem.

Pada waktunya, ahli logika yang berpendidikan akan ingin belajar setidaknya sedikit tentang berbagai jenis sistem pembuktian; minimal, Anda akhirnya harus memahami bagaimana mereka masing-masing bekerja dan menghargai keterkaitan di antara mereka. Tapi di sini - seperti biasa ketika memulai logika matematika; kita melihat khususnya pada logika aksiomatik dan satu gaya sistem deduksi alami.

(g) Pada titik ini, kemudian, kita akan memiliki dua cara yang sangat berbeda untuk mendefinisikan apa yang membuat argumen deduktif yang baik dalam logika klasik proposisional.

Kami mengatakan bahwa satu set premisses Γ secara tautologis memerlukan kesimpulan A iff setiap penilaian yang mungkin yang membuat Γ semua benar membuat A benar. (Itu ide yang didefinisikan secara semantik).

Kita sekarang juga dapat mengatakan bahwa Γ menghasilkan kesimpulan A dalam sistem bukti yang Anda pilih S jika ada derivasi tipe S dari kesimpulan A dari premisses di Γ. (Ini adalah masalah ada array bukti dengan bentuk sintaksis yang tepat).

Tentu saja, kami ingin kedua pendekatan ini cocok bersama. Kami ingin sistem bukti S favorit kami menjadi suara - seharusnya tidak memberikan positif palsu. Dengan kata lain, jika ada turunan S dari A dari Γ, maka A benar-benar secara tautologis disyaratkan oleh Γ. Kami juga ingin sistem bukti S favorit kami menjadi lengkap Γ - kami ingin menangkap semua klaim entailment semantik yang benar. Dengan kata lain, jika A secara tautologis disyaratkan oleh himpunan premisses Γ, maka memang ada beberapa turunan S dari A dari premisses di Γ.

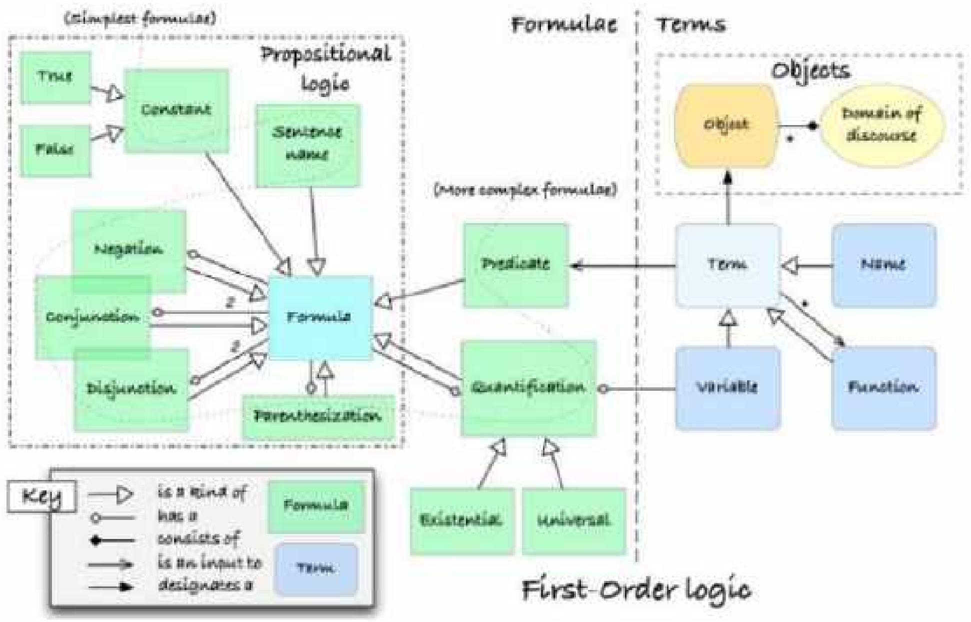
Jadi, singkatnya, kita ingin menetapkan kelengkapan sistem bukti S untuk logika proposisional (aksiomatik, deduksi alami, apa pun). Dalam membangun kelengkapan untuk logika proposisional Anda akan perlu menemukan beberapa strategi yang berguna yang nantinya dapat ditingkatkan untuk memberikan hasil kelengkapan untuk logika yang lebih kuat.

Kesimpulan

Logika proposisional dan preposisi adalah komponen integral dari algoritma pemrograman, memberikan landasan logis yang diperlukan untuk desain dan implementasi algoritma yang efektif. Mereka memberdayakan programmer untuk bernalar tentang nilai-nilai kebenaran pernyataan, membuat keputusan berdasarkan informasi, dan mengontrol aliran eksekusi kode. Dengan memanfaatkan konsep-konsep ini, programmer dapat membuat algoritma yang tidak hanya benar tetapi juga kuat, efisien, dan cocok untuk memecahkan berbagai masalah komputasi.

4 KONJUNGSI LOGIS, PERNYATAAN MAJEMUK, DAN TABEL KEBENARAN

Peta Konsep



Gambar 4.1 Peta konsep konjungsi logis, pernyataan majemuk, dan tabel kebenaran (Alspaugh, 2019)

Pendahuluan

Penghubung Logis

Kata-kata atau kelompok kata seperti "dan", atau, jika ... kemudian, jika dan hanya jika, tidak" digunakan untuk menggabungkan atau menghubungkan dua atau lebih kalimat sederhana (Crewe, 1990). Kata-kata penghubung ini disebut penghubung logis.

Pernyataan majemuk

Pernyataan baru yang dibentuk dengan menggabungkan dua atau lebih pernyataan sederhana dengan menggunakan penghubung logis disebut pernyataan majemuk (Caron et al., 1988).

Pernyataan Komponen

Pernyataan sederhana yang digabungkan menggunakan penghubung logis disebut pernyataan komponen.

Misalnya:

Pertimbangkan pernyataan sederhana berikut,

1. c adalah vokal
2. b adalah konsonan

Kedua pernyataan komponen ini dapat digabungkan dengan menggunakan penghubung logis "atau" seperti yang ditunjukkan di bawah ini:

"C adalah vokal atau B adalah konsonan."

Pernyataan di atas disebut **pernyataan majemuk yang** dibentuk dengan menggunakan penghubung logis "atau".

Tabel Kebenaran

Nilai kebenaran pernyataan majemuk tergantung pada nilai kebenaran pernyataan komponennya (Copi et al., 2016).

Penghubung Logis

a. DAN [∧] *Conjuction*

Jika p dan q adalah dua pernyataan yang dihubungkan oleh kata "dan", maka pernyataan majemuk yang dihasilkan "p dan q" disebut konjungsi p dan q yang ditulis dalam bentuk simbolik sebagai "$p \wedge q$" (Rao, 2002).

Misalnya:

p: Hari ini adalah hari yang menyenangkan.
q: Saya ingin pergi berbelanja.

Konjungsi dari dua pernyataan di atas adalah "$p \wedge q$" yaitu "Hari ini adalah hari yang menyenangkan dan saya ingin pergi berbelanja".

∧ Konjungsi benar jika dan hanya jika p dan q benar.

Tabel kebenaran untuk konjungsi p dan q adalah seperti yang ditunjukkan di bawah ini:

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

CATATAN: Kata-kata seperti tetapi, namun, masih, terlepas, meskipun, dan terlebih lagi juga digunakan untuk menghubungkan pernyataan sederhana. Kata-kata ini umumnya digunakan dengan mengganti "dan"

b. ATAU |v| (*D isjuction*)

Jika p dan q adalah dua pernyataan yang dihubungkan oleh kata "atau", maka pernyataan majemuk yang dihasilkan "p atau q" disebut disjungsi p dan q yang dalam bentuk simbolik sebagai "p v q" (Adams, 1965).

Kata "atau" digunakan dalam bahasa Inggris dalam dua pengertian yang berbeda, eksklusif dan inklusif.

Misalnya:

1. Rahul akan lulus atau gagal ujian.
2. Kandidat harus lulusan atau pasca sarjana.

Dalam misalnya (1), "atau" menunjukkan bahwa hanya satu dari dua kemungkinan yang ada tetapi tidak keduanya yang disebut arti eksklusif "atau". Dalam misalnya (2), "atau" menunjukkan bahwa kemungkinan pertama atau kedua atau kedua mungkin ada yang disebut arti inklusif "atau".

v Disjungsi salah hanya jika p dan q salah.

Tabel kebenaran untuk disjungsi p dan q adalah seperti yang ditunjukkan di bawah ini:

p	q	p v q
T	T	T
T	F	T
F	T	T
F	F	F

LATIHAN 1

1. Ungkapkan pernyataan berikut dalam bentuk simbolis:
   a. Mangga adalah buah, tetapi kentang adalah sayuran.
   b. Entah kita bermain sepak bola atau pergi bersepeda.
   c. Susu berwarna putih atau rumput berwarna hijau.
   d. Terlepas dari cacat fisik, Rahul pertama berdiri di kelas.
   e. Jagdish tinggal di rumah sementara Shrijeet dan Shalmali pergi untuk menonton film.

2. Tulislah nilai-nilai kebenaran dari pernyataan-pernyataan berikut.
   a. $\sqrt{3}$ adalah bilangan rasional atau 3 + i adalah bilangan kompleks.
   b. Jupiter adalah sebuah planet dan Mars adalah bintang.
   c. $2+3 \neq 5$ atau $2 \times 3 < 5$.
   d. $2 \times 0 = 2$ dan $2 + 0 = 2$.
   e. 9 adalah kuadrat sempurna tetapi 11 adalah bilangan prima.
   f. Moskow adalah Rusia atau London di Prancis.

c. TIDAK |~| (*Negation*)

Jika p adalah pernyataan apa pun, negasi dari p yaitu, "bukan p" dilambangkan dengan ~p (Wittgenstein et al., 1992). Negasi dari setiap pernyataan sederhana p juga dapat dibentuk dengan menulis "Tidak benar bahwa" atau "Itu salah bahwa", sebelum hal.

Misalnya:

p: Mangga adalah buah
~p: Mangga bukanlah buah

p	~p
T	F
F	T

Catatan: Jika suatu pernyataan benar, negasinya salah dan sebaliknya.

LATIHAN 2

Tuliskan negasi dari pernyataan berikut:

1. Roma berada di Italia.
2. 5 + 5 = 10
3. 3 lebih besar dari 4.
4. Jonhn pandai arung jeram (*river rafting*).
5. x adalah bilangan irasional.
6. Kuadrat bilangan real adalah positif.
7. Nol bukanlah bilangan kompleks.
8. Re (z) ≤ |z|.
9. Matahari terbenam di Timur.
10. Tidak benar bahwa mangga tidak mahal.

d. JIKA... MAKA (Implikasi, → (Bersyarat)

Jika p dan q adalah dua pernyataan sederhana, maka pernyataan majemuk, "jika p maka q", yang berarti "pernyataan p menyiratkan pernyataan q atau pernyataan q tersirat oleh pernyataan p", disebut pernyataan kondisional dan dilambangkan dengan p → q atau p → q (Valadkhani, 2022).

Di sini p disebut anteseden (hipotesis) dan q disebut konsekuen (kesimpulan).

Misalnya:

p: Saya bepergian dengan kereta api.
q: Perjalanan saya akan lebih murah.

Di sini pernyataan kondisionalnya adalah

"p → q: Jika saya bepergian dengan kereta api maka perjalanan saya akan lebih murah."

Pernyataan kondisional salah jika dan hanya jika anteseden benar dan konsekuen salah.

Tabel kebenaran untuk kondisional adalah seperti yang ditunjukkan di bawah ini:

p	q	p → q
T	T	T
T	F	F
F	T	T
F	F	T

Catatan: Bentuk setara dari pernyataan kondisional

p → q:

1. p cukup untuk q.
2. q diperlukan untuk p.
3. p menyiratkan q.
4. p hanya jika q.
5. q mengikuti dari hal.

e. Converse, Inverse dan Contrapositive (Sayyad, 2020)

Pernyataan:

Jika p → q diberikan, maka itu

Converse adalah q → p

Kebalikannya adalah ~p → ~q

Kontrapositif adalah ~q → ~p

Converse adalah kebalikan dari implikasi.

Kebalikan adalah negasi implikasi.

Sementara itu, kontraposisi adalah kebalikannya dan negasi implikasi.

Misalnya:

p: Smita cerdas.
q: Smita akan bergabung dengan Medical.

1. q → p: Jika Smita bergabung dengan Medical maka dia cerdas.
2. ~p → ~q: Jika Smita tidak cerdas maka dia tidak akan bergabung dengan Medis.
3. ~q → ~p: Jika Smita tidak bergabung dengan Medical maka dia bukan cerdas.

Pertimbangkan, tabel kebenaran berikut:

p	q	p → q	~p	~q	q → p	~q → ~p	~p → ~q
T	T	T	F	F	T	T	T
T	F	F	F	T	T	F	T
F	T	T	T	F	F	T	F
F	F	T	T	T	T	T	T

Dari tabel di atas, kami menyimpulkan bahwa:

1. Pernyataan kondisional dan kontrapositifnya selalu setara.
2. Kebalikan dan kebalikan dari pernyataan kondisional selalu setara.

f. Jika dan hanya jika (Implikasi Ganda, $\leftrightarrow$)/Biconditional

Jika p dan q adalah dua pernyataan, maka "p jika dan hanya jika q" atau "p jika q" disebut pernyataan biconditional dan dilambangkan dengan $p \leftrightarrow q$. di sini, baik p dan q disebut implisan (Quine, 2009).

Misalnya:

p: Kenaikan harga
q: Permintaan turun

Di sini pernyataan Biconditional benar jika dan hanya jika kedua implikator memiliki nilai kebenaran yang sama.

Tabel kebenaran untuk biconditional adalah seperti yang ditunjukkan di bawah ini:

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

LATIHAN

1. Ekspresikan hal-hal berikut dalam bentuk simbolis.

a. Saya suka bermain tetapi tidak bernyanyi.

b. Anand tidak suka kriket atau tenis.

c. Rekha dan Rama adalah saudara kembar.

d. Tidak benar bahwa "1" adalah bilangan real.

e. Baik 25 adalah persegi atau 41 dapat dibagi dengan 7.

f. Rani tidak pernah bekerja keras namun dia mendapat nilai bagus.

g. Meskipun tidak mendung; Masih hujan.

2. Jika p: Gadis itu senang, q: Gadis itu sedang bermain, ungkapkan kalimat berikut dalam bentuk simbolis.

A. Entah gadis-gadis itu bahagia, atau mereka tidak bermain.

b. Gadis-gadis itu tidak bahagia, tetapi mereka bermain.

c. Tidak benar bahwa gadis-gadis itu tidak bermain tetapi bahagia.

3. Temukan nilai kebenaran dari pernyataan berikut.
   a. 14 adalah bilangan komposit atau 15 adalah bilangan prima.
   b. Baik 21 adalah bilangan prima maupun tidak dapat dibagi dengan 3.
   c. Tidak benar bahwa $4 + 3i$ adalah bilangan real.
   d. 2 adalah satu-satunya bilangan prima genap dan 5 membagi 26.
   e. Baik 64 adalah kuadrat sempurna atau 46 adalah bilangan prima.
   f. $3 + 5 > 7$ jika dan hanya jika $4 + 6 < 10$.

4. Nyatakan konvers, inverse (kebalikan), dan kontrapositif dari pernyataan kondisional berikut:
   a. Jika hujan, maka pertandingan akan dibatalkan.
   b. Jika suatu fungsi dapat dibedakan maka itu kontinu.
   c. Jika luas permukaan berkurang, maka tekanan meningkat.
   d. Jika urutan dibatasi, maka itu konvergen.

5. Jika p dan q benar dan r dan s adalah pernyataan salah, cari nilai kebenaran dari pernyataan berikut:

   a. $(p \wedge q) \vee r$

   b. $p \wedge (r \rightarrow s)$

   c. $(p \vee s) \leftrightarrow (q \wedge r)$

   d. $\sim(p \wedge \sim r) \vee (\sim q \vee s)$

6. Jika p: Ini siang hari, q: Hangat

   Berikan pernyataan majemuk dalam bentuk verbal yang dilambangkan dengan
   a. $p \wedge \sim q$
   b. $p \vee q$
   c. $p \rightarrow q$
   d. $q \leftrightarrow p$

7. Siapkan tabel kebenaran untuk hal-hal berikut:

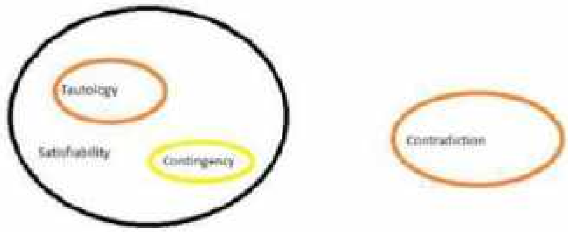
   a. $\sim p \wedge q$

   b. $p \rightarrow (p \vee q)$

   c. $\sim p \leftrightarrow q$

5 POLA PERNYATAAN DAN TAUTOLOGI KESETARAAN LOGIS, KONTRADIKSI, DAN KONTINGENSI DALAM BIDANG LOGIKA INFORMATIKA

Peta Konsep



Gambar 5.1 Peta konsep tautologi kesetaraan logis, kontradiksi, dan kontingensi (Mathematics, 2023)

Pendahuluan

Dunia logika informatika adalah domain menarik yang menggali jauh ke dalam jalinan pernyataan yang rumit, polanya, dan konsep mendalam tentang kesetaraan logis, kontradiksi, dan kontingensi. Dalam pembahasan kali ini, kami memulai eksplorasi prinsip-prinsip dasar yang mendukung esensi informasi dan komputasi.

Pada intinya, logika informatika adalah disiplin yang berusaha mengungkap struktur dan makna pernyataan yang melekat. Dalam pengejaran ini, kita menemukan banyak pola pernyataan, masing-masing dengan karakteristik dan signifikansinya yang unik. Pola-pola ini berfungsi sebagai blok bangunan penalaran logis, memungkinkan kita untuk membedakan kebenaran dari kepalsuan, dan untuk membuat keputusan berdasarkan informasi di bidang komputasi dan ilmu informasi.

Di antara konsep yang paling mendasar dalam logika informatika adalah kesetaraan logis, kontradiksi, dan kontingensi. Kesetaraan logis memungkinkan kita untuk menentukan kapan dua pernyataan setara dalam makna, menyoroti pentingnya presisi dan kejelasan dalam mengekspresikan ide-ide dalam bidang informatika. Kontradiksi, di sisi lain, menyoroti konflik dan inkonsistensi yang melekat yang mungkin timbul dalam serangkaian pernyataan, berfungsi sebagai alat penting untuk mengidentifikasi kesalahan dan menyempurnakan argumen logis. Kontingensi, pada intinya, mengingatkan kita bahwa tidak semua pernyataan secara inheren benar atau salah; Beberapa ada dalam keadaan ketidakpastian, di mana nilai kebenarannya tergantung pada faktor atau kondisi eksternal.

Sepanjang eksplorasi kami, kami akan menggali lebih dalam konsep-konsep ini, mengungkap seluk-beluk pola pernyataan dan implikasi mendalam dari kesetaraan logis, kontradiksi, dan kontingensi dalam bidang logika informatika. Bergabunglah dengan kami dalam perjalanan intelektual ini saat kami menavigasi permadani pemikiran logis yang kaya dan penerapannya di dunia informasi dan komputasi yang terus berkembang.

Pembahasan

Pola Pernyataan

Biarkan p, q, r, ... pernyataan majemuk yang diperoleh dari pernyataan sederhana ini dan dengan menggunakan satu atau lebih penghubung $\wedge$, $\vee$, ~, $\rightarrow$, $\leftrightarrow$ disebut pola pernyataan (Hamilton, 1988; Jeffrey & Burgess, 2006).

Poin-poin berikut harus dicatat saat menyiapkan tabel kebenaran dari pola pernyataan:

a. Tanda kurung harus diperkenalkan di mana pun diperlukan.
   Misalnya:
   ~(p $\wedge$ q) dan ~p $\wedge$ q tidak sama.
b. Jika pola pernyataan terdiri dari pernyataan "n" dan penghubung "m", maka tabel kebenaran terdiri dari 2 baris dan kolom (m+n).

Kesetaraan Logis

Dua pernyataan logis dikatakan setara jika dan hanya jika nilai kebenaran di kolom masing-masing dalam tabel kebenaran bersama identik (Copi, 2018; Quine, 2009).

Jika S_1 dan S_2 adalah pola pernyataan yang setara secara logis, maka dapat ditulis.

S_1-S_2

Misalnya:

Untuk membuktikan: p $\wedge$ q - ~(p $\rightarrow$ ~q)

p	q	p $\wedge$ q	~q	(p$\rightarrow$ ~q)	~(*p* $\rightarrow$ ~q)
T	T	T	F	F	T
T	F	F	T	T	F
F	T	F	F	T	F
F	F	F	T	T	F

Dalam tabel kebenaran di atas, semua entri dalam kolom p $\wedge$ q dan ~(p $\rightarrow$ ~q) identik.

$\therefore$ p $\wedge$ q - ~(p $\rightarrow$ ~q)

Catatan:

1. ~(p $\vee$ q) - ~p $\wedge$ ~q (Hukum De-Morgan ke-1) (Brzozowski, 2000; De Morgan, 2019).

					Identik	
p	q	~p	~q	p $\vee$ q	~(p $\vee$ q)	~*p* $\wedge$~*q*
T	T	F	F	T	F	F
T	F	F	T	T	F	F
F	T	T	F	T	F	F
F	F	T	T	F	T	T

2. $\sim(p \wedge q) - \sim p \vee \sim q$ (Hukum De-Morgan ke-2) (De Morgan, 1882).

					Identik	
p	q	$\sim p$	$\sim q$	$p \wedge q$	$\sim(p \wedge q)$	$\sim p \vee \sim q$
T	T	F	F	T	F	F
T	F	F	T	F	T	T
F	T	T	F	F	T	T
F	F	T	T	F	T	T

3. $p \rightarrow q - (\sim p) \vee q$

			Identik	
p	q	$\sim p$	$p \rightarrow q$	$\sim p \vee q$
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

4. $p \leftrightarrow q - (p \rightarrow q) \wedge (q \rightarrow p)$

				Identik	
p	q	$p \rightarrow q$	$q \rightarrow p$	$p \leftrightarrow q$	$(p \rightarrow q) \wedge (q \rightarrow p)$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

Tautologi, Kontradiksi dan Kontingensi

Tautologi

Pola pernyataan yang memiliki nilai kebenaran selalu T, terlepas dari nilai kebenaran dari pernyataan komponennya disebut Tautologi (Simintiras, 2000; Suppes, 1999).

Misalnya, pertimbangkan $(p \leftrightarrow q) \leftrightarrow (q \leftrightarrow p)$

p	q	$p \leftrightarrow q$	$q \leftrightarrow p$	$(p \leftrightarrow q) \leftrightarrow (q \leftrightarrow p)$
T	T	T	T	T
T	F	F	F	T
F	T	F	F	T
F	F	T	T	T

Dalam tabel kebenaran di atas, semua entri di kolom terakhir adalah T.

∴ Pola pernyataan yang diberikan adalah tautologi.

Kontradiksi

Pola pernyataan yang memiliki nilai kebenaran selalu F, terlepas dari nilai kebenaran dari pernyataan komponennya disebut Kontradiksi.

Misalnya, perhatikan $p \wedge \sim p$

p	$\sim p$	$p \wedge \sim p$
T	F	F
F	T	F

Dalam tabel kebenaran di atas, semua entri di kolom terakhir adalah F.

$\therefore$ Pola pernyataan yang diberikan adalah kontradiksi.

Kontingensi

Pola negarawan yang bukan tautologi atau kontradiksi disebut Kontingensi.

p	q	$p \leftrightarrow q$	$\sim q$	$p \rightarrow q\sim$	$\sim(p \rightarrow \sim q)$	$(p \leftrightarrow q) \wedge \sim (p \rightarrow \sim q)$
T	T	T	F	F	T	T
T	F	F	T	T	F	F
F	T	F	F	T	F	F
F	F	T	T	T	F	F

Dalam tabel kebenaran di atas, entri di kolom terakhir adalah kombinasi dari T dan F.

$\therefore$ Pola pernyataan yang diberikan bukanlah tautologi atau kontaminasi, itu adalah kontingensi.

LATIHAN

1. Persiapkan tabel kebenaran dari pola pernyataan berikut:
   a. $|(p \rightarrow q) \wedge q| \rightarrow p$
   b. $(p \wedge q) \rightarrow (\sim p)$
   c. $(p \rightarrow q) \leftrightarrow (\sim p \vee q)$
   d. $(p \leftrightarrow r) \wedge (q \leftrightarrow p)$
   e. $(p \vee \sim q) \rightarrow (r \wedge p)$

2. Dengan menggunakan tabel kebenaran, buktikan kesetaraan logis berikut:
   a. $(p \wedge q) - \sim (p \rightarrow \sim q)$
   b. $p \leftrightarrow q - (p \wedge q) \vee (\sim p \wedge \sim q)$
   c. $(p \wedge q) \rightarrow r - p \rightarrow (q \rightarrow r)$
   d. $p \vee (q \wedge r) - (p \vee q) \wedge (p \vee r)$

3. Dengan menggunakan tabel kebenaran, periksa apakah pola pernyataan berikut adalah tautologi, kontradiksi atau kontingensi.

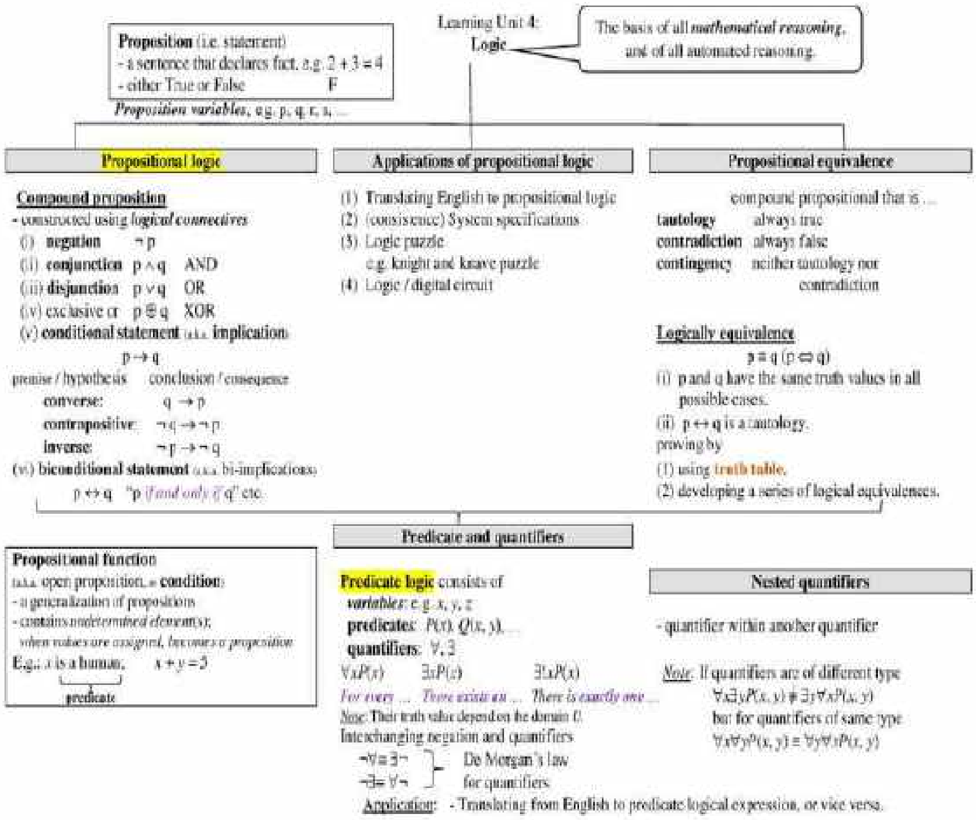
   a. $(p \wedge \sim q) \leftrightarrow (p \rightarrow q)$
   b. $(\sim p \wedge q) \wedge (q \rightarrow p)$
   c. $(p \wedge q) \vee (p \wedge r)$
   d. $|(p \vee q) \vee r| \leftrightarrow |p \vee (q \vee r)|$
   e. $(p \vee q) \wedge (p \vee r)$

Kesimpulan

Logika informatika terus mendorong batas-batas apa yang mungkin dalam ranah informasi dan komputasi. Kejelasan, presisi, dan penalaran yang masuk akal tetap menjadi kekuatan pendorong, membimbing kita menuju inovasi yang membentuk dunia digital. Dalam menghadapi kompleksitas yang terus meningkat, prinsip-prinsip yang dieksplorasi dalam pembahasan ini memberi kita alat untuk menavigasi logika yang rumit dan membuka jalan bagi masa depan dimana informasi dan perhitungan dimanfaatkan untuk potensi penuh mereka.

6 PENGUKURAN DAN PERNYATAAN UKURAN

Peta Konsep



Gambar 6.1 Peta konsep pengukuran dan pernyataan ukuran (Chu, 2020)

Pendahuluan

Quantifier (pengukuran) dan quantified statements (peryataan pengukuran) adalah konsep dasar dalam logika matematika dan penalaran formal yang memainkan peran penting dalam berbagai bidang, termasuk matematika, ilmu komputer, filsafat, dan linguistik (Johnson, 2021; Miller, 2020; Roberts, 2017; Smith, 2022). Konsep-konsep ini digunakan untuk menyatakan sejauh mana kondisi atau properti tertentu berlaku dalam domain atau set tertentu. Quantifier memungkinkan kita untuk membuat pernyataan yang tepat tentang jumlah objek yang memenuhi kondisi tertentu.

Ada dua jenis utama quantifier:

Universal Quantifier (∀): Quantifier universal, sering diwakili oleh simbol "∀," menegaskan bahwa pernyataan berlaku untuk semua elemen dalam domain tertentu (Davis, 2018). Dengan kata lain, ini mengukur semua anggota himpunan, menyatakan bahwa properti atau **kondisi tertentu** berlaku untuk setiap elemen dalam himpunan itu (Thompson, 2020).

Quantifier Eksistensial (∃): Quantifier eksistensial, sering dilambangkan dengan simbol "∃," menegaskan bahwa setidaknya ada satu elemen dalam domain tertentu yang memenuhi kondisi tertentu (Anderson, 2019). Ini mengukur keberadaan objek yang memenuhi kriteria yang ditentukan.

Pernyataan terkuantifikasi digunakan untuk mengekspresikan proposisi, pernyataan, atau hipotesis matematika dengan cara yang tepat dan tidak ambigu. Mereka adalah bagian penting dari bukti matematika, di mana quantifier membantu menetapkan kebenaran atau kepalsuan pernyataan berdasarkan sifat-sifat elemen yang dipertimbangkan.

Misalnya, pernyataan "Untuk semua bilangan asli n, n adalah genap" dapat direpresentasikan menggunakan kuantifikasi universal sebagai ∀ n (n adalah genap), sedangkan pernyataan "Ada bilangan asli n sedemikian rupa sehingga n adalah prima" dapat dinyatakan menggunakan kuantifikasi eksistensial sebagai ∃ n (n adalah prima).

Kuantifikasi dan pernyataan terkuantifikasi adalah alat utama untuk penalaran dan pembuktian teorema dalam berbagai sistem matematika dan logis, dan mereka adalah dasar untuk diskusi tentang himpunan, hubungan, fungsi, dan banyak lagi (Harris, 2019). Memahami konsep-konsep ini sangat penting bagi siapa saja yang terlibat dalam penalaran formal, matematika, atau ilmu komputer, karena mereka menyediakan kerangka kerja yang kuat untuk membuat klaim yang tepat dan ketat tentang sifat-sifat objek dan hubungannya (Lee, 2018).

Pembahasan

Kuantifikasi/Pengukuran/Quantifier

Quantifier adalah simbol yang digunakan untuk menunjukkan sekelompok kata atau frasa (White, 2021). Secara umum, dua jenis quantifier digunakan. Mereka adalah sebagai berikut:

1. Kuantifikasi Universal
   Simbol "∀" adalah singkatan dari "semua nilai" dan dikenal sebagai quantifier universal.
   Misalnya, pertimbangkan $A = \{1, 2, 3\}$
   Biarkan p: $\forall x \in A, x < 4$
   Di sini, pernyataan p menggunakan quantifier "untuk semua" (∀)
   Pernyataan ini benar jika dan hanya jika masing-masing dan setiap elemen himpunan A memenuhi kondisi "$x < 4$" dan sebaliknya salah.
   Di sini, pernyataan yang diberikan berlaku untuk semua elemen himpunan A, karena 1, 2, 3 memenuhi kondisi, '$x \in A, x < 4$'

2. Kuantifikasi eksistensial

   Simbol "∃" adalah singkatan dari "ada" dan dikenal sebagai quantifier eksistensial.

   Misalnya, pertimbangkan A = {4, 14, 66, 70}.

   Misalkan p: ∃ x ∈ A sedemikian rupa sehingga x adalah angka ganjil.

   Di sini, pernyataan yang diberikan salah karena tidak ada elemen himpunan A yang memenuhi kondisi, x ∈ A sedemikian rupa sehingga x adalah bilangan ganjil.

Pernyataan Terkuantifikasi

Pernyataan yang berisi quantifier dikenal sebagai pernyataan quantified. Umumnya, kalimat terbuka dengan quantifier menjadi pernyataan dan disebut pernyataan quantified.

Misalnya:

Gunakan quantifier untuk mengonversi kalimat terbuka **$x + 2 < 4$** menjadi pernyataan.

Jawaban: $\exists\, x \in N$ sedemikian rupa sehingga $x + 2 < 4$, adalah pernyataan yang benar, karena $x = 1 \in N$ memenuhi $x + 2 < 4$.

LATIHAN

1. Jika A = {3, 4, 6, 8}, tentukan nilai kebenaran dari masing-masing hal berikut:
   a. $\exists\, x \in A$, sehingga $x + 4 = 7$
   b. $\forall\, x \in A$, $x + 4 < 10$
   c. $\forall\, x \in A$, $x + 5 \geq 13$
   d. $\exists\, x \in A$, sehingga x ganjil
   e. $\exists\, x \in A$, sehingga $(x - 3) \in N$

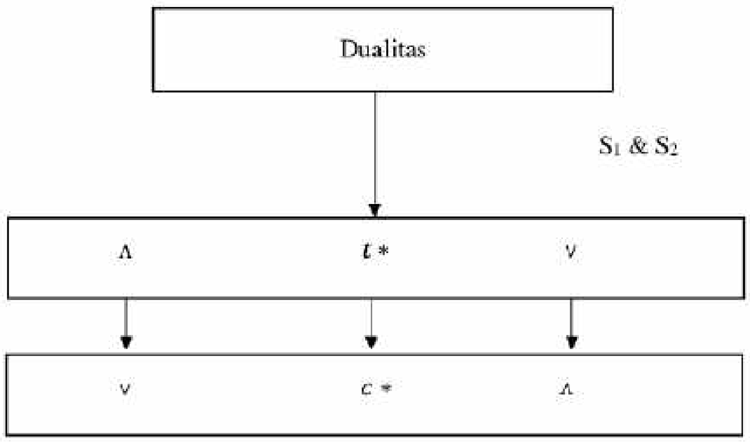
2. Gunakan quantifier untuk mengonversi setiap kalimat terbuka berikut yang didefinisikan pada N, menjadi pernyataan yang benar: N = {1, 2, 3, 4, 5, ...}
   a. $x^2 = 25$
   b. $2x + 3 < 15$
   c. $x - 3 = 11$
   d. $x^2 + 1 \leq 5$
   e. $x^2 - 3x + 2 = 0$

Kesimpulan

Pemahaman yang kuat tentang quantifier dan pernyataan quantified sangat penting bagi siapa saja yang terlibat dalam penalaran formal, matematika, atau ilmu komputer. Konsep-konsep ini memberdayakan individu untuk membuat klaim yang tepat dan tidak ambigu tentang sifat-sifat objek dan hubungan mereka, membentuk landasan pemikiran logis dan wacana matematika. Apakah Anda menjelajahi dasar-dasar matematika, mempelajari metode formal dalam ilmu komputer, atau mempelajari analisis linguistik, pemahaman tentang quantifier dan pernyataan quantified adalah kunci untuk memajukan pemahaman dan kemampuan Anda di bidang ini.

7 DUALITAS

Peta Konsep



Nota: $t*$ = Tautologi
$c*$ = Kontradiksi

Gambar 1. Peta konsep dualitas

Pendahuluan

Dualitas adalah konsep yang menemukan aplikasi dan signifikansinya diberbagai disiplin ilmu, mulai dari matematika dan fisika hingga ekonomi dan filsafat (Anderson, 2019; Davis, 2018; Johnson, 2020; Smith, 2021). Pada intinya, dualitas mengacu pada hubungan unik antara dua aspek atau perspektif sistem yang tampaknya berbeda, dimana perubahan atau transformasi dalam satu aspek sesuai dengan perubahan di aspek lainnya. Gagasan dualitas ini sering mengungkapkan hubungan yang mengejutkan dan mendalam, memberi solusi baru pada masalah yang kompleks dan memfasilitasi pemahaman yang lebih mendalam.

Dalam berbagai konteks, sudut pandang dan sebagai alat yang ampuh untuk menyederhanakan masalah yang kompleks, memberikan sudut pandang alternatif, dan meningkatkan kemampuan kita untuk memecahkan masalah yang rumit. Apakah itu dualitas antara partikel dan gelombang dalam fisika kuantum, dualitas ekonomi penawaran dan permintaan, atau dualitas filosofis pikiran dan materi, konsep ini adalah lensa yang melaluinya kita dapat memperoleh wawasan dan perspektif baru tentang sifat realitas dan hubungan antara berbagai elemen sistem.

Dalam pengantar dualitas ini, kita akan mengeksplorasi beberapa contoh umum dualitas diberbagai bidang dan menyelidiki bagaimana konsep ini digunakan untuk menyederhanakan, menyatukan, dan memperkaya pemahaman kita tentang fenomena kompleks. Dari dualitas cahaya dalam fisika hingga dualitas hak dan tanggung jawab dalam etika, kita akan melakukan perjalanan melalui berbagai manifestasi konsep yang menarik dan menyatukan ini.

Pembahasan

Dualitas

Dua pernyataan majemuk S_1 dan S_2 dikatakan ganda satu sama lain jika salah satu dapat diperoleh dari yang lain **dengan menukar** "$\wedge$" **dan** "$\vee$" dan sebaliknya (Davis, 2018; Harris, 2019; Johnson, 2020; Lee, 2020; Roberts, 2018; Smith, 2021). Penghubung **"$\wedge$" dan "$\vee$"** adalah ganda satu sama lain. Juga, ganda diperoleh dengan mengganti t dengan c dan c dengan t, di mana "t" menunjukkan tautologi dan "c" menunjukkan kontradiksi.

Catatan:

1. **Simbol "$\sim$" tidak berubah saat adanya dual.**
2. **Dual dari dualitas adalah pernyataan itu sendiri.**
3. **Pernyataan khusus "t" (tautologi) dan "c" (kontradiksi) adalah dual satu sama lain.**
4. **T diubah menjadi F dan sebaliknya**.

Prinsip dualitas

Jika pernyataan majemuk S_1 hanya berisi $\sim$, $\wedge$, dan $\vee$ dan S_2 muncul dari S_1 dengan mengganti $\wedge$ oleh $\vee$ dan $\vee$ oleh $\wedge$, maka S_1 adalah tautologi jika dan hanya jika S_2 adalah kontradiksi.

LATIHAN

1. Tuliskan ganda dari pernyataan majemuk berikut:
   a. $(p \wedge q) \vee r$
   b. $T \vee (p \vee q)$
   c. $p \wedge [\sim q \vee (p \wedge q) \vee \sim r]$

JAWAB

a. $(p \vee q) \wedge r$
b. $F \wedge (p \wedge q)$
c. $p \vee [\sim q \wedge (p \vee q) \wedge \sim r]$

2. Tuliskan pernyataan ganda dari masing-masing pernyataan majemuk berikut:
   a. Vijay dan Vinay tidak bisa berbahasa Inggris.
   b. Ravi atau Avinash pergi ke Chennai.
   c. Madhuri memiliki rambut keriting dan mata cokelat.

JAWAB

a. Vijay atau Vinay tidak bisa berbahasa Inggris.
b. Ravi dan Avinash pergi ke Chennai.
c. Madhuri memiliki rambut keriting atau mata cokelat.

3. Tulislah dualitas dari pernyataan berikut:
   a. $(p \vee q) \vee r - p \vee (q \vee r)$
   b. $p \vee (q \wedge r) - (p \vee q) \wedge (p \vee r)$

JAWAB

a. $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
b. $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

4. Tuliskan dualitas dari masing-masing pernyataan berikut dimana t adalah tautologi dan c adalah kontradiksi.
   a. $p \wedge q \wedge c$
   b. $\sim p \wedge (q \vee c)$
   c. $(p \wedge t) \vee (c \wedge \sim q)$

JAWAB

a. $p \vee q \vee t$
b. $\sim p \wedge (q \vee c)$
c. $(p \wedge t) \vee (c \wedge \sim q)$

Kesimpulan

Dualitas berfungsi sebagai konsep yang luar biasa dan serbaguna yang berguna diberbagai disiplin ilmu dan bidang pemahaman manusia. Ini adalah lensa yang melaluinya, kita mendapatkan perspektif baru tentang sistem dan masalah yang kompleks. Dengan menyoroti keterkaitan aspek atau sudut pandang yang tampaknya berbeda, dualitas menyederhanakan, menyatukan, dan memperdalam pemahaman kita tentang dunia.

Dualitas memberdayakan kita untuk menjembatani kesenjangan antara ide-ide yang tampaknya berlawanan, mengungkapkan koneksi tersembunyi dan menyelaraskan unsur-unsur yang tampaknya kontradiktif. Dari sifat ganda cahaya dalam fisika hingga dualitas ekonomi penawaran dan permintaan, dan bahkan dualitas filosofis yang mendalam antara pikiran dan materi, konsep ini menunjukkan bagaimana berbagai aspek realitas saling berhubungan dan saling bergantung.

Nilai dualitas terbukti dalam aplikasi praktisnya diberbagai bidang, dimana ia sering mengarah pada solusi inovatif, wawasan yang lebih mendalam, dan kemampuan pemecahan masalah yang ditingkatkan. Ketika kita terus mengeksplorasi seluk-beluk alam semesta, dari ranah mikrokosmik fisika partikel ke domain makrokosmik ekonomi dan etika, dualitas tetap menjadi alat mendasar untuk mendapatkan pemahaman yang lebih komprehensif dan holistik tentang dunia kita yang kompleks.

8 NEGASI PERNYATAAN MAJEMUK

Peta Konsep

Gambar 8.1 Peta konsep negasi pernyataan majemuk

Pendahuluan

Konsep negasi dari pernyataan majemuk adalah aspek fundamental dari logika dan kalkulus proposisional. Ini melibatkan proses pembentukan kebalikan logis atau penolakan dari pernyataan kompleks yang menggabungkan beberapa proposisi menggunakan operator logis seperti "dan" (konjungsi), "atau" (disjungsi), "tidak" (negasi), "jika-maka" (implikasi), dan "jika dan hanya jika" (biconditional) (Atlas, 1997; Bendall, 1979; García-Madruga et al., 2001; Handley et al., 2006; Hoosain, 1973; Tanduk & Wansing, 2015; Khemlani et al., 2012, 2014; Wansing, 2017).

Dalam logika, pernyataan majemuk adalah kombinasi dari dua atau lebih pernyataan sederhana, yang dikenal sebagai proposisi. Negasi dari pernyataan majemuk, sering disebut sebagai "negasi konjungsi," "negasi disjungsi," atau "negasi implikasi," berusaha untuk membalikkan nilai kebenaran dari seluruh pernyataan. Proses ini sangat penting untuk menilai validitas argumen, memecahkan masalah logis, dan membuat keputusan berdasarkan informasi.

Negasi pernyataan majemuk tergantung pada jenis operator logis yang digunakan dalam pernyataan. Misalnya, negasi konjungsi (A dan B) setara dengan disjungsi negasi proposisi komponennya (¬A atau ¬B), dan negasi implikasi (A → B) setara dengan konjungsi anteseden dan negasi konsekuen (A dan ¬B).

Memahami cara meniadakan pernyataan majemuk dengan benar adalah keterampilan mendasar dalam penalaran logis dan pemikiran kritis, karena memungkinkan individu untuk menganalisis proposisi yang kompleks, menilai nilai kebenarannya, dan mengevaluasi validitas argumen logis. Pengetahuan ini berharga tidak hanya dalam konteks matematika dan filosofis tetapi juga dalam pengambilan keputusan sehari-hari dan skenario pemecahan masalah di mana pemikiran yang jelas dan tepat diperlukan.

Pembahasan

Negasi pernyataan majemuk

1. Negasi konjungsi
   Negasi dari konjungsi dua pernyataan sederhana adalah disjungsi negasi mereka. $\sim(p \wedge q) — \sim p \vee \sim q$

2. Negasi disjungsi
   Negasi dari disjungsi dua pernyataan sederhana adalah konjungsi dari negasi mereka.
   $\sim(p \vee q) — \sim p \wedge \sim q$

3. Negasi negasi
   Negasi negasi dari pernyataan sederhana adalah pernyataan itu sendiri.
   Jika p adalah pernyataan sederhana, maka $\sim(\sim p) — p$

4. Negasi pernyataan kondisional (implikasi)
   Negasi dari pernyataan kondisional $p \rightarrow q$ adalah p tetapi bukan q.
   $\sim(p \rightarrow q) — p \wedge \sim q$

5. Negasi pernyataan biconditional (implikasi ganda)
   Negasi dari pernyataan biconditional $p \leftrightarrow q$ adalah negasi dari $p \rightarrow q$ atau $p \rightarrow p$.
   $\sim(p \leftrightarrow q) — (p \wedge \sim q) \vee (q \wedge \sim p)$

   Catatan:
   Negasi pola pernyataan yang melibatkan satu atau lebih pernyataan sederhana p, q, r, ... dan satu atau lebih penghubung ~, ∧, ∨ adalah diperoleh dengan mengganti ∨ dengan ∧ dan mengganti ∧ dengan ∨. Pernyataan p, q, r digantikan oleh negasinya ~p, ~q, ~r. Begitu juga sebaliknya.

6. Negasi pernyataan terkuantifikasi

Sementara apabila menemukan negasi dari pernyataan terkuantifikasi, kata "semua" maka digantikan dengan "beberapa" dan "untuk setiap" digantikan oleh "ada" dan sebaliknya.

LATIHAN

a. Semua segitiga sama sisi sama kaki.
b. Beberapa bilangan real bukanlah bilangan kompleks.
c. Setiap siswa telah membayar biaya.
d. $\forall x \in N, n + 1 > 2$.
e. $\forall x \in N, x^2 + x$ adalah bilangan genap.
f. $\exists n \in N$, sehingga $n^2 = n$.
g. $\exists x \in R$, sehingga $x^2 = x$.
h. Semua mahasiswa perguruan tinggi ini tinggal di asrama.
i. Beberapa fungsi kontinu dapat dibedakan.
j. Demokrasi bertahan jika para pemimpinnya tidak korup.
k. Kondisi yang diperlukan dan cukup bagi seseorang untuk menjadi sukses adalah jujur.
l. Beberapa persamaan kuadrat memiliki akar yang tidak sama.

JAWAB

a. Beberapa segitiga sama sisi tidak sama kaki.
b. Semua bilangan real adalah bilangan kompleks.
c. Beberapa siswa belum membayar biaya.
d. $\exists n \in N$, sehingga $n + 1 \leq 2$.
e. $\exists x \in R$, sehingga $x^2 = x$ bukan bilangan genap.
f. $\forall n \in N, n^2 \neq n$
g. $\forall x \in R, x^2 \geq x$.
h. Beberapa mahasiswa perguruan tinggi ini tidak tinggal di asrama.
i. Semua fungsi kontinu tidak dapat dibedakan.
j. Biarkan, p : Para pemimpin tidak korup.
   q : Demokrasi bertahan.

$\therefore$ Pernyataan yang diberikan adalah dari bentuk, $p \rightarrow q$ negasinya adalah dari bentuk $p \wedge \sim q$

Sehinga menjadi "Para pemimpin tidak korup, tetapi demokrasi tidak bertahan"

k. Misalkan, p : Seseorang sukses.
   q : Seseorang jujur.

$\therefore$ Pernyataan yang diberikan adalah dari bentuk, $p \rightarrow q$ negasinya adalah dari bentuk

$(p \wedge \sim q) \vee (q \wedge \sim p)$

Sehinga menjadi "Seseorang sukses, tetapi dia tidak jujur atau seseorang jujur, tetapi dia tidak sukses"

l. Semua persamaan kuadrat memiliki akar yang sama.

5. Dengan menggunakan aturan negasi, tulis negasi berikut ini.
   a. $p \wedge (q \rightarrow r)$
   b. $(\sim p \vee q) \wedge r$
   c. $(\sim p \wedge \sim q) \vee (p \wedge \sim q)$

JAWAB

a. $\sim[p \wedge (q \rightarrow r)]$

$\equiv \sim p \;\; \vee \sim (q) \rightarrow r$ ... (Negasi konjungsi)

$\equiv)\ \sim p \vee (q \wedge \sim r$ ... (Negasi implikasi)

b. $\sim[\sim p \vee q) \wedge r)]$

$\equiv \sim(\sim p \vee q)) \vee r$ ... (Negasi konjuksi)

— $[\sim(\sim p) \wedge \sim q)] \vee \sim r$ .... (Negasi disjuction)

— $(p \wedge \sim q) \vee \sim r$ .... (Negasi dari negasi)

c. $\sim[(\sim p \wedge \sim q) \vee (p \wedge \sim q)]$ — $\sim(\sim p \wedge \sim q) \wedge \sim(p \wedge \sim q)$ ... (Negasi disjungsi)

— $[\sim(\sim p) \vee \sim(\sim q)] \wedge [(\sim p \vee \sim(\sim q)]$ ... (Negasi konjungsi)
— $(p \vee q) \wedge (\sim p \vee q)$ ... (Negasi dari negasi)

6. Bentuk negasi dari pernyataan berikut maka berikan pembenaran (justifikasi).
   a. $(p \wedge q) \rightarrow (\sim p \vee r)$
   b. $(q \vee \sim r) \wedge (p \vee q)$

JAWAB

a. $\sim[(p \wedge q) \rightarrow (\sim p \vee r)]$ — $(p \wedge q) \wedge \sim(\sim p \vee r)$ ... (Negasi implikasi)

— $(p \wedge q) \wedge [\sim(\sim p) \wedge \sim r)]$ ... (Negasi disjungsi)

— $(p \wedge q) \wedge (p \wedge \sim r)$ ... (Negasi dari negasi)

b. $\sim[(q \vee \sim r) \wedge (p \vee q)]$ — $\sim(q \vee \sim r) \vee \sim(p \vee q)$ ... (Negasi konjungsi)

— $[\sim q \wedge \sim(\sim r)] \vee (\sim p \wedge \sim q)$ ... (Negasi disjungsi)

— $(\sim q \wedge r) \vee (\sim p \wedge \sim q)$ ... (Negasi dari negasi)

— $(\sim q \wedge r) \vee (\sim q \wedge \sim p)$ ... (Hukum komunikatif)
— $\sim q \wedge (r \vee \sim p)$ ... (Hukum distributif)

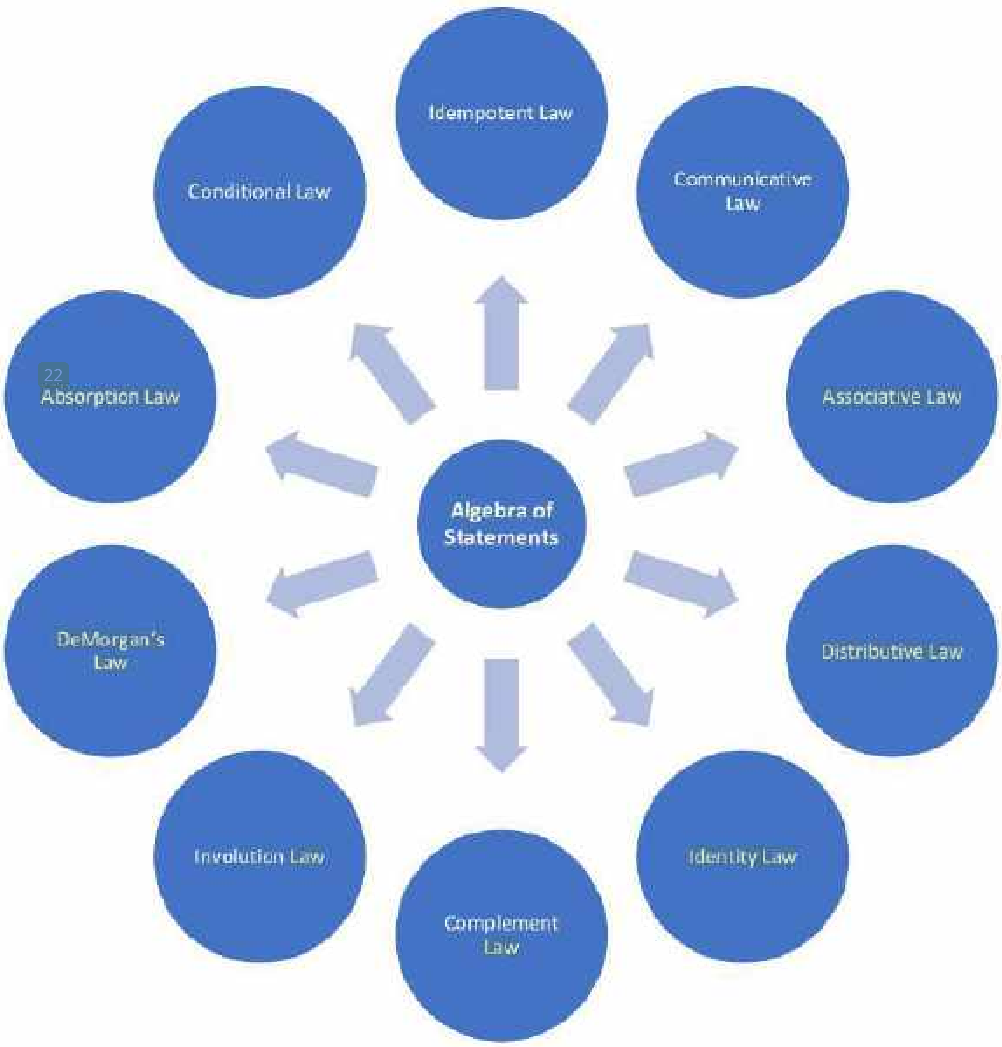
Kesimpulan

Konsep "Negasi dari pernyataan majemuk" adalah prinsip dasar dan mendasar dalam logika dan kalkulus proposisional. Meskipun mungkin tidak memiliki referensi spesifik atau sumber khusus, ia memainkan peran penting dalam analisis dan manipulasi pernyataan logis.

Negasi dari pernyataan majemuk melibatkan pembentukan kebalikan logis atau penolakan dari pernyataan kompleks yang menggabungkan beberapa proposisi menggunakan operator logis. Proses ini sangat penting untuk menilai validitas argumen, memecahkan masalah logis, dan membuat keputusan berdasarkan informasi. Ini memungkinkan kita untuk memeriksa hubungan antara proposisi, nilai-nilai kebenarannya, dan bagaimana mereka berinteraksi dalam konstruksi logis yang kompleks. Memahami negasi pernyataan majemuk adalah keterampilan mendasar dalam penalaran logis dan pemikiran kritis, dengan aplikasi dalam matematika, filsafat, ilmu komputer, dan berbagai bidang di mana analisis yang tepat dan logis diperlukan.

9 ALJABAR PERNYATAAN (BEBERAPA PERNYATAAN STANDAR YANG SETARA)

Peta Konsep



Gambar 9.1 Peta konsep aljabar pernyataan

Pendahuluan

Dalam matematika dan logika, struktur aljabar sering melibatkan operasi dan hubungan antar elemen. Jika kita mempertimbangkan pernyataan atau proposisi, yang merupakan ekspresi yang benar atau salah, "Aljabar Pernyataan" mungkin melibatkan operasi yang menggabungkan atau memanipulasi pernyataan ini dengan cara yang analog dengan operasi aljabar.

Misalnya, dalam logika proposisional, yang berhubungan dengan hubungan logis antara proposisi, Anda memiliki operasi logis seperti konjungsi (∧), disjungsi (∨), dan negasi (¬). Operasi ini memungkinkan Anda untuk membentuk pernyataan majemuk dari yang lebih sederhana.

Pembahasan

Aljabar Pernyataan

Beberapa pernyataan setara standar:

1. Hukum Idempoten: (Kolokoltsov & Maslov, 1997).
   a. $p \vee q$ — p
   b. $p \wedge q$ — p

2. Hukum Komunikatif: (Mode, 1945).
   a. $p \vee q$ — $q \vee p$
   b. $p \wedge q$ — $q \wedge p$

3. Hukum Asosiatif: (Suschkewitsch, 1929).
   a. $(p \vee q) \vee r$ — $p \vee (q \vee r)$ — $p \vee q \vee r$
   b. $(p \wedge q) \wedge r$ — $p \wedge (q \wedge r)$ — $p \wedge q \wedge r$

4. Hukum Distributif: (Appelgate et al., 1969).
   a. $p \vee (q \wedge r)$ — $(p \vee q) \wedge p \vee r$
   b. $p \wedge (q \vee r)$ — $(p \wedge q) \vee p \wedge r$

5. Hukum Identitas: (Shoenfield, 2018).
   a. $p \vee F$ — p
   b. $p \wedge F$ — F
   c. $p \vee T$ — T
   d. $p \wedge T$ — p

6. Hukum Pelengkap: (Moss, 2011).
   a. $p \vee \sim p$ — T
   b. $p \wedge \sim p$ — F

7. Hukum Involusi: (Moreno & Budesca, 2000).
   a. $\sim T$ — F
   b. $\sim F$ — T

c. $\sim(\sim p) — p$

8. Hukum DeMorgan: (Johnstone, Agustus).
   a. $\sim(p \vee q) — \sim p \wedge \sim q$
   b. $\sim(p \wedge q) — \sim p \vee \sim q$

9. Hukum Absorpsi: (Visser, 2021).
   a. $p \vee (p \wedge q) — p$
   b. $p \wedge (p \vee q) — p$

10. Hukum Bersyarat: (Guzman & Squier, 1990).
   a. $p \rightarrow q — \sim p \vee q$
   b. $p \leftrightarrow q — (\sim p \vee q) \wedge (\sim q \vee p)$

Idempotent laws	$p \vee p \equiv p$	$p \wedge p \equiv p$
Associative laws	$(p \vee q) \vee r \equiv p \vee (q \vee r)$	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
Commutative laws	$p \vee q \equiv q \vee p$	$p \wedge q \equiv q \wedge p$
Distributive laws	$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$	$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
Identity laws	$p \vee F \equiv p \quad p \vee T \equiv T$	$p \wedge F \equiv F \quad p \wedge T \equiv p$
Involution laws	$\neg\neg p \equiv p$	
Complement laws	$\neg p \vee p \equiv T$	$\neg p \wedge p \equiv F$
DeMorgan's laws	$\neg(p \wedge q) \equiv \neg p \vee \neg q$	$\neg(p \vee q) \equiv \neg p \wedge \neg q$
Conditional identities	$p \rightarrow q \equiv \neg p \vee q$	$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

LATIHAN

1. Tanpa menggunakan tabel kebenaran, tunjukkan itu.
   a. $p \wedge (q \vee \sim p) — p \wedge q$
   b. $(p \wedge q) \vee (\sim p \wedge q) \vee (\sim q \wedge r) — q \vee r$

JAWAB

a. L.H.S= $p \wedge (q \vee \sim p) \equiv (p \wedge q) \vee (p \wedge \sim p) \ldots$ (Hukum distributif)
$\equiv (p \wedge q) \vee F \ldots$ (Hukum pelengkap)
$\equiv (p \vee F) \wedge (q \vee F)$ (Hukum distributif)
$\equiv p \wedge q \ldots$ (Hukum identitas)

$\therefore$ L.H.S= R.H.S

$\therefore p \wedge (q \vee \sim p) — p \wedge q$

b. L.H.S $= (p \wedge q) \vee (\sim p \wedge q) \vee (\sim q \wedge r) \equiv [(p \vee \sim p) \wedge q] \vee (\sim q \wedge r)$… (Hukum Asosiatif dan Distributif)

$\equiv (T \wedge q) \vee (\sim q \wedge r)$ … (Hukum pelengkap)

$\equiv q \vee (\sim q \wedge r)$ … (Hukum identitas)

$\equiv (q \vee \sim q) \wedge (q \vee r)$ … (Hukum distributif)

$\equiv (T \wedge (q \vee r)$ … (Hukum pelengkap)

$\equiv q \vee r$ … (Hukum identitas)

$\therefore$ L.H.S= R.H.S

$\therefore (p \wedge q) \vee (\sim p \wedge q) \vee (\sim q \wedge r)$ — $q \vee r$

Kesimpulan

Aljabar pernyataan mengacu pada kerangka matematika atau logis yang melibatkan manipulasi dan analisis pernyataan, kemungkinan akan mencakup seperangkat hukum atau aturan analog dengan yang ditemukan dalam aljabar Boolean. Undang-undang ini mungkin mengatur operasi pada pernyataan logis, memberikan prinsip-prinsip untuk penyederhanaan, manipulasi, dan analisis.

Dalam arti yang lebih luas, setiap sistem aljabar yang diterapkan pada pernyataan kemungkinan akan melibatkan aturan yang terkait dengan kombinasi, transformasi, dan penyederhanaan ekspresi logis. Sistem seperti itu akan menjadi dasar dalam bidang-bidang seperti logika matematika, ilmu komputer, dan filsafat, di mana manipulasi pernyataan dan proposisi adalah aspek fundamental.

10 PENERAPAN LOGIKA UNTUK SWITCHING CIRCUIT

Pendahuluan

Penerapan logika untuk beralih sirkuit merupakan persimpangan penting antara bidang elektronik dan penalaran formal. Dalam konteks ini, logika, pendekatan sistematis untuk penalaran, dimanfaatkan untuk merancang dan menganalisis sirkuit switching, yang membentuk blok bangunan mendasar dari sistem digital. Perkawinan logika dan sirkuit switching telah berperan dalam evolusi komputasi modern, memungkinkan penciptaan perangkat elektronik kompleks yang memproses, menyimpan, dan mengirimkan informasi.

Pada intinya, penerapan logika untuk switching circuit melibatkan pemanfaatan operasi logis — seperti AND, OR, dan NOT — untuk memanipulasi sinyal dalam bentuk biner. Bahasa biner ini adalah dasar dari komputasi digital dan memungkinkan kontrol komponen elektronik yang tepat. Dengan menggunakan prinsip-prinsip logis, insinyur dapat merancang sirkuit yang melakukan fungsi tertentu, membuka jalan bagi pengembangan komputer digital, mikroprosesor, dan sistem elektronik canggih lainnya.

Pentingnya aplikasi ini melampaui pertimbangan teoritis, karena memiliki implikasi praktis yang mendalam. Kemampuan untuk secara sistematis mewakili dan memanipulasi informasi menggunakan operasi logis dalam sirkuit switching telah merevolusi industri mulai dari telekomunikasi hingga otomatisasi. Dampaknya terbukti dalam teknologi sehari-hari, dari smartphone dan laptop hingga sistem kontrol industri, di mana integrasi logika dan sirkuit switching yang mulus memainkan peran penting.

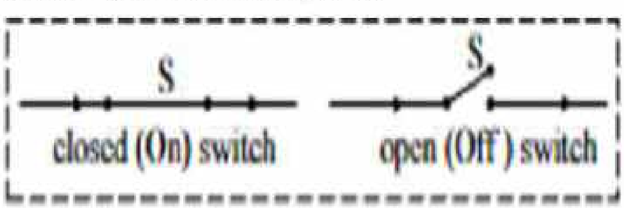
Pembahasan

Penerapan Logika untuk Switching (saklar) Circuit

Kerja saklar listrik mirip dengan pernyataan logis yang memiliki tepat dua hasil, yaitu, T atau F (Kohavi & Jha, 2009; Muroga, 1997; Sasao, 2012). Switch juga memiliki dua hasil atau hasil (arus mengalir dan arus tidak mengalir) tergantung pada status switch yaitu (ON atau OFF). Analogi ini sangat berguna dalam memecahkan masalah perancangan rangkaian dengan bantuan logika (McCluskey, 1986; Roth et al., 2020; Straubing, 2012).

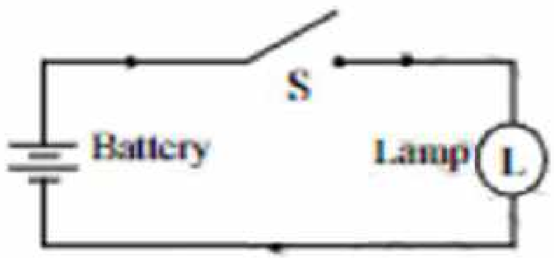
Sakelar

Saklar listrik adalah perangkat dua keadaan yang digunakan untuk menghidupkan arus "on" atau "off" (Matsuo & Harada, 1976).



Seperti yang ditunjukkan pada gambar di atas, jika sakelar menyala yaitu, sirkuit ditutup, arus melewati sirkuit dan sebaliknya.

Pertimbangkan sirkuit sederhana yang memiliki sakelar "S"; baterai dan lampu "L". Ketika sakelar "S" ditutup (mis., ON, arus mengalir melalui sirkuit), lampu bersinar (menyala). Demikian pula, ketika saklar terbuka (yaitu, OFF, arus tidak mengalir melalui sirkuit), lampu tidak menyala (mati).



Jadi, jika p adalah pernyataan 'saklar ditutup' dan jika *l adalah pernyataan 'lampu bersinar' maka p setara dengan* 1 yaitu p — *l*.

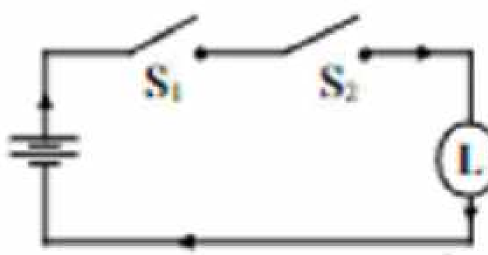
Catatan:

1. ~p berarti 'sakelar terbuka'. Dalam hal ini, lampu tidak akan menyala dan dengan demikian ~p — ~*l*.

2. Jika saklar adalah 'ON' maka nilai kebenarannya adalah T atau 1 **dan jika saklar adalah 'OFF', nilai kebenarannya adalah** F atau 0 (Palem, 2005).

Jika ada dua sakelar, maka mereka dapat dihubungkan dengan cara berikut:

1. Switch S_1 dan S_2 terhubung secara seri (Hinago & Koizumi, 2011):



Misalkan p: sakelar S_1 ditutup
q: saklar S_2 ditutup
l: lampu bersinar

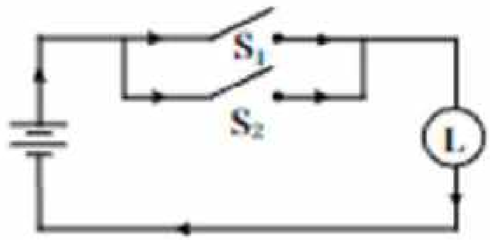
Dalam hal ini, lampu menyala, jika dan hanya jika kedua sakelar ditutup.

Jadi, kita memiliki, $p \wedge q$ — *l*

Tabel input – output (switching):

p	q	p ∧ q
T (1)	T (1)	T (1)
T (1)	F (0)	F (0)
F (0)	T (1)	F (0)
F (0)	F (0)	F (0)

2. Switch S_1 dan S_2 sejajar (Tsunoda et al., 2013).



Biarp: sakelar S_1 ditutup
q: saklar S_2 ditutup
l: lampu bersinar

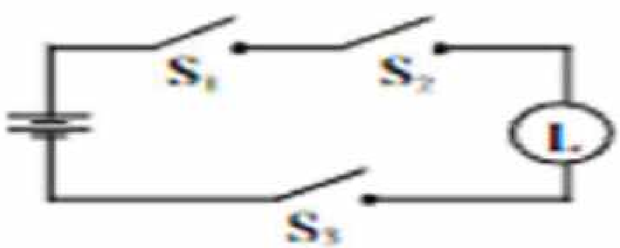
Dalam hal ini, lampu menyala, jika setidaknya salah satu sakelar ditutup.

Jadi, kita memiliki $p \vee q — l$

Tabel input – output (switching):

p	q	p ∨ q
T (1)	T (1)	T (1)
T (1)	F (0)	T (1)
F (0)	T (1)	T (1)
F (0)	F (0)	F (0)

Kedua jaringan di atas dapat digabungkan untuk membentuk jaringan yang rumit seperti gambar di bawah ini:



Biarp: sakelar S_1 ditutup
q: saklar S_2 ditutup
r: Sakelar S_3 ditutup
l: lampu bersinar

Dalam hal ini, lampu menyala, jika S_1 dan S_2 keduanya ditutup atau jika S_3 ditutup.
Jadi, kita memiliki, $(p \wedge q) \vee r — l$

Catatan:

a. Jika dua atau lebih switch dalam rangkaian terbuka atau tertutup secara bersamaan, maka mereka dilambangkan dengan huruf yang sama dan disebut 'switch **setara**.'

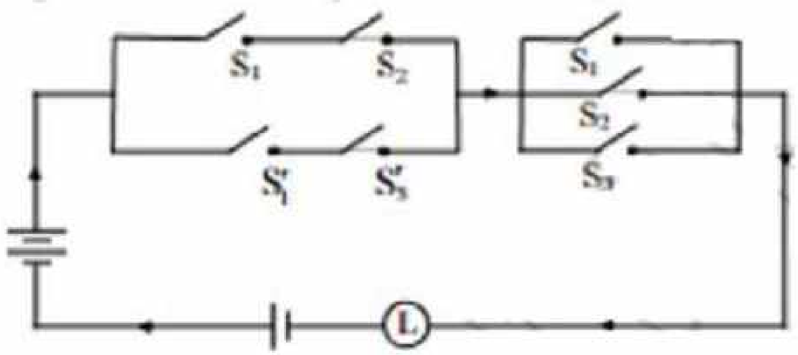
b. Setiap dua saklar dalam rangkaian yang memiliki keadaan berlawanan disebut saklar **komplementer**.
Misalnya, jika S_1 dan S_2 adalah dua switch sedemikian rupa sehingga ketika S_1 ditutup, S_2 terbuka dan sebaliknya, maka switch S_1 dan S_2 disebut S_1'. Dalam situasi seperti itu, salah satunya dianggap sebagai p dan yang lainnya sebagai ~p atau p'.

c. Dua sirkuit disebut **setara** jika output dari dua sirkuit selalu sama.

d. Rangkaian disebut lebih **sederhana** jika berisi jumlah sakelar yang lebih sedikit.

Contoh:

Ekspresikan sirkuit berikut dalam bentuk simbolis:



Solusi:

Misalkan p: sakelar S_1 ditutup
q: saklar S_2 ditutup
r: Sakelar S_3 ditutup
~p: sakelar ditutup S_1'
~r : sakelar ditutupS_3'
l: lampu L bersinar 'menyala'

Lampu L 'menyala' jika dan hanya jika —

i. Switch S_1 dan Switch S_2 ditutup.
($\therefore$ S_1 dan S_2 seri)

atau ii. Switch S_1' dan S_3' ditutup.
($\therefore$ S_1'dan S_3' seri)

dan iii. Switch S_1 atau S_2 atau S_3 ditutup.

(∴ S_1 atau S_2 atau S_3 adalah paralel)

∴ Bentuk simbolis dari rangkaian yang diberikan adalah,

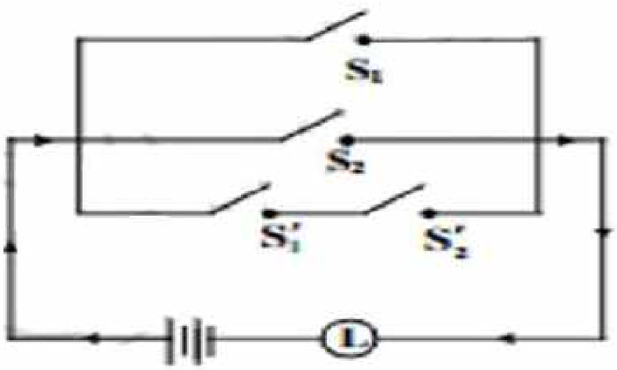
$[(p \wedge q) \vee (\sim p \wedge \sim r)] \wedge (p \vee q \vee r) — l$

Umumnya, l tidak ditulis dan oleh karena itu bentuk simboliknya adalah

$[(p \wedge q) \vee (\sim p \wedge \sim r)] \wedge (p \vee q \vee r)$

LATIHAN

1. Tulislah simbolis dan tabel input-output atau switching dari sirkuit berikut.
   a.



JAWAB

Misalkaan p: Sakelar S_1 ditutup
q: Switch S_2 ditutup
~p: Sakelar S_1' ditutup atau sakelar S_1 terbuka.
~q: Sakelar S_2' ditutup atau sakelar S_2 terbuka.

∴ Bentuk simbolis dari rangkaian yang diberikan adalah,
$(p \vee q) \vee (\sim p \wedge \sim q)$

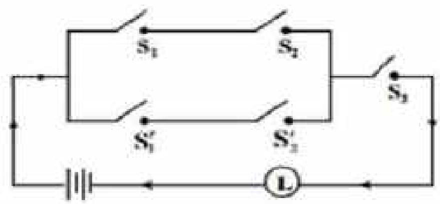
Tabel input-output:

p	q	~p	~q	p ∨ q	~p ∧ ~q	(p ∨ q) ∨ (~p ∧ ~q)
1	1	0	0	1	0	1
1	0	0	1	1	0	1
0	1	1	0	1	0	1
0	0	1	1	0	1	1

Tabel Switching:

p	q	~p	~q	p ∨ q	~p ∧ ~q	(p ∨ q) ∨ (~p ∧ ~q)
T	T	F	F	T	F	T
T	F	F	T	T	F	T
F	T	T	F	T	F	T
F	F	T	T	F	T	T

b.



JAWAB

Misalkan p: Sakelar S_1 ditutup
q: Switch S_2 ditutup
r: Sakelar S_3 ditutup
~p: Sakelar S'_1 ditutup atau sakelar S_1 terbuka.
~q: Sakelar S'_2 ditutup atau sakelar S_2 terbuka.

∴ Bentuk simbolis dari rangkaian yang diberikan adalah,
$[(p \wedge q) \vee (\sim p \wedge \sim q)] \wedge r$

Misalkan:
a —(p ∧ q)
b — (~p ∧ ~q)

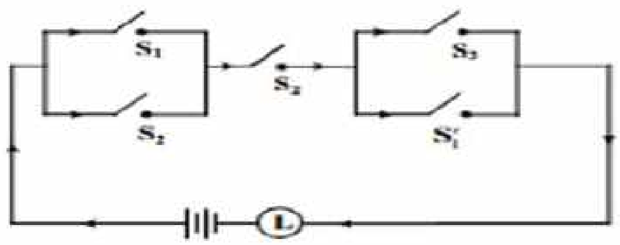
Tabel input-output:

p	q	r	~p	~q	a	b	a ∨ b	(a ∨ b) ∧ r
1	1	1	0	0	1	0	1	1
1	1	0	0	0	1	0	1	0
1	0	1	0	1	0	0	0	0
1	0	0	0	1	0	0	0	0
0	1	1	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0
0	0	1	1	1	0	1	1	1
0	0	0	1	1	0	1	1	0

Tabel Switching:

p	q	r	~p	~q	a	b	a ∨ b	(a ∨ b) ∧ r
T	T	T	F	F	T	F	T	T
T	T	F	F	F	T	F	T	F
T	F	T	F	T	F	F	F	F
T	F	F	F	T	F	F	F	F
F	T	T	T	F	F	F	F	F
F	T	F	T	F	F	F	F	F
F	F	T	T	T	F	T	T	T
F	F	F	T	T	F	T	T	F

c.



JAWAB

Misalkan p: Sakelar S_1 ditutup
q: Switch S_2 ditutup
r: Sakelar S_3 ditutup
~p: Sakelar S'_1 ditutup atau sakelar S_1 terbuka.

∴ Bentuk simbolis dari rangkaian yang diberikan adalah,
$(p \vee q) \wedge q \wedge (r \vee \sim p)$

Tabel input-output:

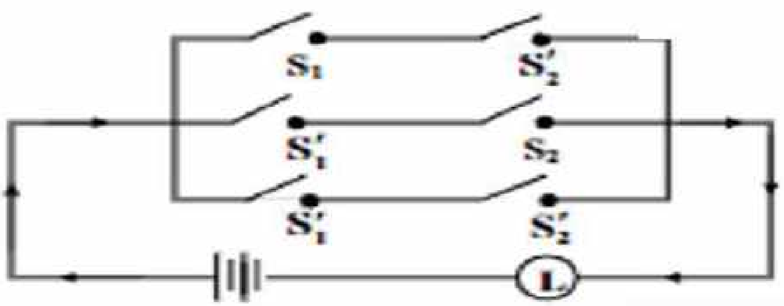
p	q	r	~p	p ∨ q	(p ∨ q) ∧ q	(r ∨~p)	(p ∨ q) ∧ q ∧ (r ∨ ~p)
1	1	1	0	1	1	1	1
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0
1	0	0	0	1	0	0	0
0	1	1	1	1	1	1	1
0	1	0	1	1	1	1	1
0	0	1	1	0	0	1	0
0	0	0	1	0	0	1	0

Tabel Switching:

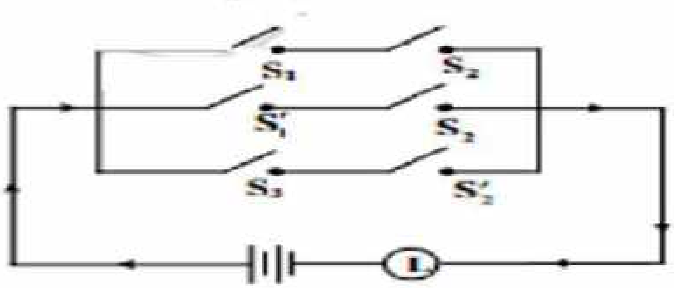
p	q	r	~p	p ∨ q	(p ∨ q) ∧ q	(r ∨ ~p)	(p ∨ q) ∧ q ∧ (r ∨ ~p)
T	T	T	F	T	T	T	T
T	T	F	F	T	T	F	F
T	F	T	F	T	F	T	F
T	F	F	F	T	F	F	F
F	T	T	T	T	T	T	T
F	T	F	T	T	T	T	T
F	F	T	T	F	F	T	F
F	F	F	T	F	F	T	F

2. Buat sirkuit switching dari pernyataan berikut.
   a. $(p \wedge q \wedge r) \vee \sim p \vee (q \wedge \sim r)$

3. Berikan pengaturan alternatif untuk rangkaian berikut, sehingga rangkaian barunya hanya memiliki dua saklar. Juga tulis tabel switchingnya.



4. Selesaikan aspek berikut:
   a. Bentuk simbolis,
   b. Tabel Switching, dan
   c. Sirkuit switching yang disederhanakan



Kesimpulan

Penerapan logika untuk beralih sirkuit membentuk landasan elektronik digital, memfasilitasi penciptaan sistem rumit yang telah mendefinisikan kembali cara kita memproses dan berinteraksi dengan informasi. Persimpangan logika dan teknologi ini terus membentuk lanskap komputasi modern, menggarisbawahi peran penting yang dimainkan oleh penalaran formal dalam evolusi sistem elektronik.

11 PENYEDERHANAAN

Pendahuluan

Penyederhanaan dalam logika matematika adalah proses menyusun atau mengekspresikan pernyataan logis secara lebih sederhana atau lebih ringkas. Tujuan dari penyederhanaan ini adalah untuk membuatnya lebih mudah untuk memahami, menganalisis, dan memanipulasi pernyataan logis. Beberapa teknik umum yang digunakan dalam menyederhanakan logika matematika melibatkan hukum logika dan aljabar.

Pembahasan

Pernyataan aljabar

1. Hukum Idempoten: (Kolokoltsov & Maslov, 1997).
   a. $p \vee q$ — p
   b. $p \wedge q$ — p

2. Hukum Komunikatif: (Mode, 1945).
   a. $p \vee q$ — $q \vee p$
   b. $p \wedge q$ — $q \wedge p$

3. Hukum Asosiatif: (Suschkewitsch, 1929).
   a. $(p \vee q) \vee r$ — $p \vee (q \vee r)$ — $p \vee q \vee r$
   b. $(p \wedge q) \wedge r$ — $p \wedge (q \wedge r)$ — $p \wedge q \wedge r$

4. Hukum Distributif: (Appelgate et al., 1969).
   a. $p \vee (q \wedge r)$ — $(p \vee q) \wedge p \vee r$
   b. $p \wedge (q \vee r)$ — $(p \wedge q) \vee p \wedge r$

5. Hukum Identitas: (Shoenfield, 2018).
   a. $p \vee F$ — p
   b. $p \wedge F$ — F
   c. $p \vee T$ — T
   d. $p \wedge T$ — p

6. Hukum Pelengkap: (Moss, 2011).
   a. $p \vee \sim p$ — T
   b. $p \wedge \sim p$ — F

7. Hukum Involusi: (Moreno & Budesca, 2000).
   a. $\sim T$ — F
   b. $\sim F$ — T

c. $\sim(\sim p) \equiv p$

8. Hukum DeMorgan: (Johnstone, Agustus).
   a. $\sim(p \vee q) \equiv \sim p \wedge \sim q$
   b. $\sim(p \wedge q) \equiv \sim p \vee \sim q$

9. Hukum Absorpsi: (Visser, 2021; Guzman & Squier, 1990).
   a. $p \vee (p \wedge q) \equiv p$
   b. $p \wedge (p \vee q) \equiv p$

10. Hukum Bersyarat: (Guzman & Squier, 1990).
    a. $p \rightarrow q \equiv \sim p \vee q$
    b. $p \leftrightarrow q \equiv (\sim p \vee q) \wedge (\sim q \vee p)$

Penyederhanaan

Penyederhanaan juga dapat digunakan untuk membuktikan kesetaraan atau kesamaan logis. Untuk menyederhanakan hal-hal, hal pertama yang harus dihapus adalah $\rightarrow$ dan $\leftrightarrow$ dan membuat kombinasi $\wedge$, $\vee$, dan $\sim$. Beberapa contoh kesamaan logis.

$p \rightarrow q \equiv (\sim p \vee q)$

$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p) \equiv (\sim p \vee q) \wedge (\sim q \vee p) \equiv (p \wedge q) \vee (\sim p \wedge \sim q)$

Operasi penyederhanaan menggunakan hukum logika dapat digunakan untuk membuktikan tautologi jika hasil akhirnya adalah T (1), kontradiksi jika hasilnya adalah F (0), dan kontingen jika hasil akhirnya adalah T dan F.

Coupler dasar juga disebut coupler yang cukup. Ketiga perangkat ini membentuk gerbang yang menjadi dasar dari sistem digital. Coupler hanya menunjukkan bahwa coupler $\wedge$ dapat diganti dengan $\sim$ dan $\vee$, sedangkan coupler $\vee$ dapat diganti dengan $\sim$ dan $\wedge$.

$\sim (p \wedge \sim p) \equiv \sim p \vee \sim(\sim p) \equiv \sim p \vee p$

Sederhanakan bentuk logika berikut ke dalam bentuknya yang paling sederhana:

1. $p \wedge (\sim p \rightarrow p)$

$\equiv p \wedge (\sim p \vee p)$...	(hukum impilkasi)
$\equiv p \wedge (p \vee \sim p)$...	(hukum idempoten)
$\equiv p \wedge (\sim p \vee p)$...	(hukum komunikatif)
$\equiv p \wedge (p \vee \sim p)$...	(hukum idempoten)
$\equiv p \wedge (T)$...	(hukum komplemen)
$\equiv p \wedge T$	
$\equiv p$ ...	(hukum identiti)

2. $\sim(\sim p \wedge (q \vee \sim q))$

$\equiv \sim(\sim p \wedge (q \vee \sim q))\ldots$ (hukum involusi)

$\equiv p \wedge \sim (q \vee \sim q)\ldots$ (hukum komplemen)

$\equiv p \wedge \sim (T)$

$\equiv p \wedge F \ldots$ (hukum identiti)

$\equiv F$

3. Tunjukkan bahwa: $(p \rightarrow q) \wedge (q \rightarrow p) \equiv (p \wedge q) \vee (\sim p \wedge \sim q)$

$$\equiv (\sim p \vee q) \wedge (\sim q \vee p)$$
$$\equiv (q \vee \sim p) \wedge (p \vee \sim q)$$
$$\equiv (p \vee \sim q) \wedge (q \vee \sim p)$$
$$\equiv \big((p \vee \sim q) \wedge q\big) \vee \big((p \vee \sim q) \wedge \sim p\big)$$
$$\equiv \big((p \wedge q) \vee (\sim q \wedge q)\big) \vee \big((p \wedge \sim p) \vee (\sim q \wedge \sim p)\big)$$
$$\equiv \big((p \wedge q) \vee 0\big) \vee \big(0 \vee (\sim q \wedge \sim p)\big)$$
$$\equiv (p \wedge q) \vee (\sim q \wedge \sim p)$$
$$\equiv (p \wedge q) \vee (\sim p \wedge \sim q)$$

4. Sederhanakan, Jika p, q, r, adalah pernyataan dengan nilai kebenaran T, F, T masing-masing, tentukanlah nilai-nilai kebenaran berikut:

a. $q \rightarrow (p \vee \sim r)$

b. $(\sim r \wedge p) \vee \sim q$

c. $(p \rightarrow q) \vee r$

d. $(r \wedge q) \leftrightarrow \sim p$

e. $(p \vee q) \rightarrow (q \vee r)$

JAWAB

a. $q \rightarrow (T \vee \sim T)$

$\equiv F \rightarrow (T \vee F)$

$\equiv F \rightarrow (T)$

$\equiv F \rightarrow T$

$\equiv T$

5. **Tanpa menggunakan tabel kebenaran**, buktikan bahwa $[(p \vee q) \wedge \sim p)] \rightarrow q$ itu adalah tautologi.

Cara-1

$[(p \vee q) \wedge \sim p)] \rightarrow q$

$\equiv [(p \wedge \sim p) \vee (q \wedge \sim p)] \rightarrow q$

$\equiv [(\sim q \vee \sim(\sim p)] \vee q \ldots$ (Negasi Konjuksi)

$\equiv (\sim q \vee p) \vee q$

$\equiv (p \vee \sim q) \vee q$

$\equiv p \vee (\sim q \vee q) \ldots$ (Hukum Assosiatif)

$\equiv p \vee T \ldots$ (Hukum Pelengkap)

$\equiv T \ldots$ (Hukum Identitas)

Nilai kebenaran dari pola pernyataan yang diberikan adalah T. oleh karena itu, ia adalah tautologi.

Cara-2

$(p \vee q) \wedge \sim p) \rightarrow q$

$\equiv \sim((p \vee q) \wedge \sim p) \vee q \ldots$ (Implikasi-Eliminasi)

$\equiv (\sim(p \vee q) \vee p) \vee q \ldots$ (Hukum De Morgan)

$\equiv (\sim p \wedge \sim q \vee p) \vee q \ldots$ (Hukum distributif)

$\equiv (\sim p \vee q) \vee q \ldots$ (Hukum Idempoten)

$\equiv (p \vee q) \rightarrow (\sim p \vee q) \ldots$ (Hukum implikasi)

$p \rightarrow q \equiv (\sim p \vee q)$

$\equiv T$

Nilai kebenaran dari pola pernyataan yang diberikan adalah T. oleh karena itu, ini adalah tautologi.

Kesimpulan

Penyederhanaan dalam logika matematika tidak hanya alat untuk memperpendek pernyataan, tetapi juga metode untuk meningkatkan pemahaman, efisiensi operasional, dan analisis argumen logis. Penyederhanaan membantu meningkatkan keterbacaan pernyataan logis. Dengan mengurangi ekspresi logis ke bentuk yang lebih sederhana, pernyataan menjadi lebih mudah dipahami oleh pembaca atau pengolah informasi.

Proses penyederhanaan membuat analisis logis lebih mudah. Pernyataan yang disederhanakan dapat membuatnya lebih mudah untuk melihat hubungan logis antara proposisi dan memahami struktur dasar argumen atau ekspresi logis. Dengan menyederhanakan pernyataan logis, operasi logis seperti penggabungan, penyederhanaan lebih lanjut, dan evaluasi kebenaran menjadi lebih efisien. Ini berguna dalam konteks perhitungan logis dan manipulasi simbol logis. Penyederhanaan melibatkan penerapan berbagai hukum logis seperti hukum identitas, hukum distributif, dan lain-lain. Penerapan hukum-hukum ini membantu dalam mengubah ekspresi logis menjadi bentuk yang lebih sederhana. Melalui penyederhanaan, struktur logis dari pernyataan logis dapat lebih mudah diidentifikasi. Ini membantu untuk menyoroti elemen-elemen kunci dan hubungan antara proposisi dalam sebuah argumen. Dalam konteks pembuktian matematis, penyederhanaan sering digunakan untuk membantu membangun argumen yang lebih ringkas dan mudah dipahami. Ini dapat menyederhanakan proses pembuktian teorema atau sifat tertentu.

12 KONSEP LOGIKA PREDIKATIF, INTERPRETASI, DAN ARGUMEN VALIDITAS

Pendahuluan

Eksplorasi penalaran logis dan analisis sistematis proposisi telah mengarah pada pengembangan cabang-cabang khusus dalam logika formal. Tiga konsep penting yang memainkan peran sentral dalam domain ini adalah logika predikatif, interpretasi, dan argumen validitas. Konsep-konsep ini menyediakan kerangka kerja untuk mengekspresikan pernyataan dengan variabel, membangun hubungan antara simbol abstrak dan entitas dunia nyata, dan menilai kesehatan logis argumen. Dalam diskusi ini, kita akan menyelidiki seluk-beluk logika predikatif, memeriksa perannya dalam mengartikulasikan pernyataan yang melibatkan predikat dan quantifier. Selanjutnya, kita akan mengeksplorasi pentingnya interpretasi, menjelaskan bagaimana menjembatani kesenjangan antara bahasa simbolik logika dan dunia nyata. Akhirnya, kita akan meneliti argumen validitas, mengungkap prinsip-prinsip yang membedakan penalaran logis dari pernyataan belaka. Bersama-sama, konsep-konsep ini membentuk landasan penalaran logis formal, mempengaruhi beragam bidang mulai dari matematika hingga filsafat dan ilmu komputer.

Pembahasan

Logika Predikatif

Logika predikatif adalah cabang logika formal yang berhubungan dengan predikat dan quantifier. Predikat adalah pernyataan yang melibatkan variabel dan menjadi proposisi ketika nilai-nilai tertentu diganti dengan variabel.

Kuantifier, seperti quantifier universal (∀) dan quantifier eksistensial (∃), digunakan untuk mengekspresikan pernyataan tentang semua atau beberapa elemen dalam domain tertentu.

1. Kuantifikasi Universal
   Simbol "∀" adalah singkatan dari "semua nilai" dan dikenal sebagai quantifier universal.
   Misalnya, pertimbangkan $A = \{1, 2, 3\}$
   Biarkan p: $\forall x \in A, x < 4$
   Di sini, pernyataan p menggunakan quantifier "untuk semua" (∀)
   Pernyataan ini benar jika dan hanya jika masing-masing dan setiap elemen himpunan A memenuhi kondisi "$x < 4$" dan salah sebaliknya.
   Di sini, pernyataan yang diberikan berlaku untuk semua elemen himpunan A, karena 1, 2, 3 memenuhi kondisi, '$x \in A, x < 4$'
2. Kuantifikasi eksistensial
   Simbol "∃" adalah singkatan dari "ada" dan dikenal sebagai quantifier eksistensial.
   Misalnya, pertimbangkan $A = \{4, 14, 66, 70\}$.
   Biarkan p: $\exists x \in A$ sedemikian rupa sehingga x adalah angka ganjil.

Di sini, pernyataan yang diberikan salah karena tidak ada elemen himpunan A yang memenuhi kondisi, $x \in A$ sedemikian rupa sehingga x adalah bilangan ganjil.

Logika predikatif menyediakan cara sistematis untuk mengekspresikan dan penalaran tentang pernyataan yang melibatkan variabel dan quantifier.

Interpretasi

Dalam logika, interpretasi adalah cara untuk memberikan makna pada simbol dan pernyataan dalam sistem logis. Ini mengaitkan unsur-unsur dari dunia nyata dengan variabel dan predikat bahasa logis.

Interpretasi menyediakan pemetaan antara simbol-simbol abstrak dari bahasa logis dan entitas dunia nyata. Ini membantu untuk menentukan kebenaran atau kepalsuan pernyataan dalam model yang diberikan.

Misalnya, jika kita memiliki pernyataan seperti "Untuk semua x, x lebih besar dari 0," interpretasi akan menentukan apa yang diwakili variabel x (misalnya, bilangan real), dan pernyataan itu akan benar jika, di bawah interpretasi ini, setiap bilangan real lebih besar dari 0.

Argumen yang valid dan tidak valid

Sekarang kita telah mengembangkan bahasa dasar logika, kita akan mulai mempertimbangkan bagaimana logika dapat digunakan untuk menentukan apakah argumen yang diberikan valid atau tidak.

Definisi 1.1. Argumen (bentuk) adalah urutan pernyataan (formulir). Semua pernyataan (bentuk) dalam argumen (bentuk) kecuali yang terakhir, disebut premis (atau asumsi, atau hipotesis). Pernyataan akhir (bentuk) disebut kesimpulan. Simbol $\therefore$ yang dibaca "**oleh karena itu**" biasanya ditempatkan tepat sebelum kesimpulan.

Definisi 1.2. Bentuk argumen valid jika setiap kali pernyataan yang benar diganti dengan variabel pernyataan, kesimpulannya selalu benar. Mengatakan argumen tidak valid berarti argumen tersebut tidak valid.

Poin utama mengenai argumen yang valid adalah bahwa ia mengikuti dari bentuk logis itu sendiri dan tidak ada hubungannya dengan konten. Ketika kesimpulan dicapai dengan menggunakan argumen yang valid, kami mengatakan kesimpulan disimpulkan atau disimpulkan dari premis.

Untuk menguji apakah **argumen valid atau tidak**, kami melakukan hal berikut:

(i) **Identifikasi** premis dan kesimpulannya.

(ii) Buat tabel kebenaran yang menunjukkan **nilai kebenaran** premis dan kesimpulannya

(iii) Cari semua baris di mana premis semuanya benar - kami menyebut baris **seperti itu baris kritis**. Jika kesimpulannya salah dalam baris kritis, maka argumen tersebut tidak valid. Jika tidak, argumennya valid (karena kesimpulannya selalu benar ketika premisnya benar).

Contoh:

Jika air laut surut setelah gempa bumi di laut, maka tsunami datang. Air laut surut setelah gempa bumi di laut. Itu sebabnya tsunami datang. [Valid]

Misalkan:

p: Air laut surut setelah gempa bumi di laut
q: Tsunami datang

$$\begin{array}{ll} \text{Premis 1} & p \to q \\ \text{Premis 2} & \underline{p} \\ & \therefore q \end{array}$$

Tabel kebenaran:

p	q	$p \to q$	
T	**T**	**T**	(baris 1)
T	F	F	(baris 2)
F	T	T	(baris 3)
F	F	T	(baris 4)

Dalam logika, kata-kata "benar" dan "valid" memiliki arti yang sangat berbeda - kebenaran berbicara tentang pernyataan yang membentuk argumen dan validitas berbicara tentang apakah kesimpulan mengikuti dari premis. Perhatikan bahwa argumen yang benar-benar valid mungkin memiliki kesimpulan yang salah tergantung pada nilai kebenaran premis. Demikian juga, argumen yang tidak valid mungkin memiliki kesimpulan yang benar tergantung pada nilai kebenaran premis.

Tentukan apakah argumen berikut ini valid.

Misalkan:

p: Saya banyak tidur
q: Saya tidak mengerjakan pekerjaan rumah
r: Saya akan mengerjakannya di kelas

$$\begin{array}{c} p \to q \\ q \to r \\ \hline \therefore p \vee q \to r \end{array}$$

Jika saya banyak tidur, maka saya tidak mengerjakan pekerjaan rumah saya. Jika saya tidak mengerjakan pekerjaan rumah saya, maka saya melakukannya di kelas. Oleh karena itu, jika saya banyak tidur atau tidak mengerjakan pekerjaan rumah saya, saya akan melakukannya di kelas.

Seperti disebutkan di atas, ini adalah argumen yang sangat valid, tetapi jelas bukan kesimpulan yang benar! Ini karena meskipun hipotesis pertama benar, hipotesis kedua salah (dan karena itu kesimpulannya salah – lihat tabel kebenaran).

Membangun tabel kebenaran:

p	q	r	$p \to q$	$q \to r$	$p \vee q$	$p \vee q \to r$
T	T	T	T	T	T	T
T	T	F	T	F	T	F
T	F	T	F	T	T	T
T	F	F	F	T	T	F
F	T	T	T	T	T	T
F	T	F	T	F	T	F
F	F	T	T	T	F	T
F	F	F	T	T	F	T

Sekarang kita akan mempertimbangkan beberapa bentuk argumen standar yang valid. Argumen yang valid kadang-kadang disebut **aturan inferensi** karena kesimpulan selalu dapat disimpulkan dari hipotesis. Kami akan membuat daftar argumen yang valid dan sebagian besar waktu kami tidak akan membuktikan validitas karena biasanya cukup jelas (Tabel 12.1). Setelah semuanya dinyatakan, kita akan mempertimbangkan beberapa contoh bagaimana menggunakan argumen ini.

1	Modus Ponens McGee (2018)	$p \to q$ p $\therefore q$	
2	Modus Tollens Adams (1988)	$p \to q$ $\sim q$ $\therefore \sim p$	
3	Generalization Plotkin (1970)	p $\therefore p \vee q$	q $\therefore p \vee q$
4	Conjunction Rundle (1983)	p q $\therefore p \wedge q$	
5	Specialization Andersen (1994)	$p \wedge q$ $\therefore p$	$p \wedge q$ $\therefore q$
6	Elimination Gabbay & Ohlbach (1992)	$p \vee q$ $\sim p$ $\therefore q$	$p \vee q$ $\sim q$ $\therefore p$
7	Transitivity De Baets & De Meyer (2005)	$p \to q$ $q \to r$ $\therefore p \to r$	
8	Proof by Division into Cases Hammack (2013)	$p \vee q$ $p \to r$ $q \to r$ $\therefore r$	
9	Contradiction Antonini (2018)	$\sim p \to c$ $\therefore p$	

Nama-nama aturan inferensi yang telah dijelaskan di atas biasanya menggambarkan dengan tepat apa yang dilakukan aturan inferensi. Misalnya, aturan "Eliminasi" menghilangkan salah satu pernyataan variabel yang mungkin mengingat bahwa salah satunya harus benar, dan kita tahu salah satunya tidak benar.

Kontradiksi adalah alat yang sangat penting dalam matematika dan digunakan untuk banyak bukti penting (Hanna, 1995). Namun, gagasan kontradiksi dirusak dalam kontroversi karena mengandaikan masalah penting dalam matematika (atau lebih umum, logika) yang disebut "hukum tengah yang dikecualikan". Secara khusus, hukum tengah yang dikecualikan menyatakan bahwa "p atau tidak p". Masalah utama dengan hukum tengah yang dikecualikan adalah bahwa in dapat digunakan untuk membuktikan keberadaan objek matematika tanpa pernah benar-benar membangunnya. Sebuah gerakan logika dan matematikawan yang disebut intuisionis percaya ini adalah argumen yang tidak valid karena, kecuali seseorang benar-benar dapat membangun objek matematika, maka bagaimana seseorang dapat membuktikan bahwa itu ada.

Contoh:

Telah terjadi pembunuhan di kota X. Jika motifnya adalah pencurian maka ada sesuatu yang hilang. Jika motifnya politis maka si pembunuh segera pergi. Jika Anda segera pergi, akan ada banyak jejak di TKP.

Fakta: 1. Tidak ada barang yang hilang
2. Ada banyak jejak kaki di TKP

Apa motif pembunuhan, pencurian, politik, atau sesuatu yang lain!

p: Motifnya adalah pencurian
q: Motifnya adalah politik
r: Motifnya adalah hal lain
s: Ada sesuatu yang hilang
t: Pembunuhnya segera pergi
u: Banyak jejak di ruangan itu

Aturan yang dibuat:

1. $p \to s$
2. $q \to t$
3. $u \to \sim t$

Fakta: $\sim s, u$

Penyelesaian:

1. $p \vee q \vee r$

2. $p \to s$
3. $\underline{\quad \sim s}$
$\therefore \sim p$ (Modus T)

4. Hasil No.3 & No.1 $q \vee r$

5. $u \rightarrow {\sim}t$
6. u

$\therefore \quad {\sim}t$ (Modus P)

7. $q \rightarrow t$ & hasil No. 6
8. ${\sim}t$

$\therefore {\sim}q$ (Modus T)

9. Hasil No.4 & No.8

10. $\therefore$ r Motif lain

Kesimpulan

Logika predikatif menyediakan kerangka kerja untuk mengekspresikan pernyataan yang melibatkan variabel dan quantifier, interpretasi menghubungkan pernyataan logis ini ke dunia nyata, dan validitas berkaitan dengan struktur logis argumen, memastikan bahwa kesimpulan secara logis mengikuti dari premis. Konsep-konsep ini mendasar dalam studi logika formal dan berlaku di berbagai bidang, termasuk matematika, filsafat, dan ilmu komputer.

13 DERIVASI DASAR DAN APLIKASI SEDERHANA DALAM HAL INFORMATIKA

Pendahuluan

Prinsip-prinsip derivasi dapat diterapkan dalam berbagai skenario dalam bidang informatika, menekankan pentingnya penalaran logis dalam pengembangan dan pelaksanaan proses komputasi.

Pembahasan

Derivasi, dalam konteks logika formal, berfungsi sebagai metode terstruktur **untuk menyimpulkan kesimpulan** dari premis yang diberikan (Jeffrey & Burgess, 2006). Proses mendasar ini sangat penting dalam bidang informatika, di mana penalaran yang tepat dan tidak ambigu sangat penting untuk desain dan analisis algoritma, bahasa pemrograman, dan sistem komputasi (Ghannad et al., 2019). Dalam eksplorasi dasar-dasar derivasi dan aplikasi sederhana dalam informatika ini, kami akan mengungkap prinsip-prinsip yang mengatur inferensi logis. Melalui langkah-langkah deduksi yang sistematis, derivasi memungkinkan kita untuk menarik kesimpulan yang valid, keterampilan penting dalam membangun argumen logis yang kuat dalam bidang informatika (Johnson-Laird & Byrne, 1993). Mari kita selidiki aspek dasar derivasi dan ilustrasikan penerapannya melalui sebuah contoh.

Contoh 1:

Pertimbangkan skenario berikut dalam informatika: Tugas pemrograman yang melibatkan validasi input pengguna. Misalkan kita memiliki dua premis:

Jika input pengguna adalah bilangan bulat, maka itu adalah input yang valid.

Input yang diberikan memang bilangan bulat.

Sekarang, mari kita dapatkan kesimpulan menggunakan premis ini:

Premis 1: Jika input pengguna adalah bilangan bulat, maka itu adalah input yang valid.
Premis 2: Input yang diberikan memang bilangan bulat.

Derivasi:

Mengingat Premis 2, kita dapat menggunakan Premis 1 untuk menyimpulkan bahwa input tersebut valid.

Kesimpulan: Oleh karena itu, input tersebut valid.

Contoh 2:

Pernyataan Bersyarat dalam Pemrograman (Green, 1977):

Premis 1: Jika pengguna memasukkan kata sandi yang benar.

Premis 2: Pengguna memasukkan kata sandi yang benar.

Derivasi: Oleh karena itu, pengguna diberikan akses ke sistem.

Penanganan Kesalahan dalam Suatu Fungsi (Wu et.al., 2021):

Premis 1: Jika terjadi kesalahan dalam suatu fungsi.

Premis 2: Terjadi kesalahan dalam fungsi.

Derivasi: Oleh karena itu, mekanisme penanganan kesalahan harus dipicu.

Analisis Algoritma Penyortiran (Kulalvaimozhi et al., 2015):

Premis 1: Jika suatu algoritma tidak efisien.

Premis 2: Algoritma penyortiran tidak efisien.

Derivasi: Oleh karena itu, pertimbangkan untuk menerapkan algoritma penyortiran yang lebih efisien.

Validasi Permintaan Database (Mackert & Lohman, 1986):

Premis 1: Jika kueri database benar secara sintaksis.

Premis 2: Kueri basis data secara sintaksis benar.

Derivasi: Oleh karena itu, jalankan kueri untuk mengambil data dari database.

Penanganan File dalam Sistem File (Levy & Silberschatz, 1990):

Premis 1: Jika file ada di direktori yang ditentukan.

Premis 2: File ada di direktori yang ditentukan.

Derivasi: Oleh karena itu, lanjutkan dengan membuka file untuk membaca atau menulis.

Otentikasi Pengguna dalam Pengembangan Web (Wang & Sun, 2020):

Premis 1: Jika pengguna memberikan kredensial login yang valid.

Premis 2: Pengguna memberikan kredensial login yang valid.

Derivasi: Oleh karena itu, izinkan pengguna untuk mengakses akun mereka.

Pemeriksaan Konektivitas Jaringan (Obeidat & Berkovich, 2008):

Premis 1: Jika ada koneksi internet.

Premis 2: Perangkat mendeteksi koneksi internet.

Derivasi: Oleh karena itu, aplikasi dapat melanjutkan dengan operasi online.

Kesimpulan

Dalam derivasi sederhana ini, kami menggunakan hubungan logis yang ditetapkan oleh premis untuk mendapatkan kesimpulan yang valid. Proses ini menampilkan aplikasi praktis derivasi dalam informatika, di mana penalaran logis sangat penting dalam memastikan kebenaran dan keandalan proses komputasi. Ketika kita mempelajari lebih dalam dasar-dasar derivasi, kita akan mengungkap teknik yang lebih rumit yang mendukung penalaran logis yang sehat dalam lanskap informatika yang dinamis.

14 KONSEP DASAR HIMPUNAN, OPERASI HIMPUNAN, TUPEL DASAR, URUTAN, HIMPUNAN DAYA DAN PENERAPANNYA DI BIDANG INFORMATIKA SECARA SEDERHANA DAN STUDI KASUS

Pendahuluan

Dalam dunia ilmu komputer dan informatika, konsep matematika dasar seperti himpunan, tupel, urutan, dan set daya memainkan peran penting dalam pengembangan sistem, analisis data, dan manajemen informasi. Penerapan konsep-konsep ini membantu profesional TI untuk merancang solusi yang efisien, mengatur data dengan baik, dan mengelola sumber daya informasi secara efektif.

Mengatur dan mengatur operasi

Himpunan, sebagai konsep dasar, merupakan kumpulan objek yang memiliki sifat tertentu. Dalam konteks informatika, himpunan digunakan untuk merepresentasikan dan mengelompokkan data dengan cara yang logis. Operasi-operasi seperti gabungan, irisan, selisih, dan komplemen memungkinkan pengembang untuk melakukan manipulasi data dengan mudah, yang berguna dalam konteks pengembangan aplikasi dan analisis data.

Tuple dan Sequences

Tupel, sebagai struktur data yang tidak dapat diubah, dan urutan, sebagai urutan elemen yang dapat diakses, membantu dalam mewakili dan mengelola data terstruktur. Dalam pemrograman, tupel dan urutan seperti string, daftar, dan tupel memberikan fleksibilitas dalam mewakili informasi berurutan atau terkait.

Set Daya

Konsep set daya, yang merupakan himpunan semua himpunan bagian dari suatu himpunan, memberikan landasan bagi pengelolaan struktur data yang kompleks dan analisis kompleksitas algoritma. Dalam informatika, power set digunakan dalam konteks manajemen izin, analisis daya kombinatorial, dan representasi struktur data yang kompleks.

Aplikasi di Bidang Informatika

Studi kasus penggunaan konsep-konsep ini di bidang informatika memberikan pemahaman konkret tentang bagaimana teori matematika diimplementasikan dalam pengembangan perangkat lunak, manajemen basis data, dan keamanan sistem. Dalam konteks studi kasus, kita akan mengeksplorasi bagaimana set, mengatur operasi, tupel, urutan, dan power set diimplementasikan dalam sistem manajemen database toko online, memberikan pandangan praktis pada kegunaan konsep-konsep ini dalam skenario dunia nyata.

Dengan memahami dan menerapkan konsep-konsep dasar ini, para profesional informatika dapat meningkatkan efisiensi dalam mengelola data, merancang sistem yang kuat, dan memecahkan masalah kompleks yang muncul di dunia digital saat ini.

Pembahasan

Mengeset

Satu set adalah kumpulan objek dengan properti tertentu. Objek dalam satu set disebut elemen. Set dapat diekspresikan menggunakan notasi set atau daftar elemen yang dipisahkan oleh kurung kurawal.

Karakterisasi

Satu **set** adalah kumpulan objek nol atau lebih. Set adalah kumpulan objek atau orang yang terdefinisi dengan baik.

Objek disebut **elemen** himpunan.

$a \in b$ adalah kependekan dari 'a *adalah elemen dari himpunan* b'.

Contoh

$A = \{1, 2, 3, 4, 5\}$

Himpunan bilangan bulat positif kurang dari 4:

$\{1, 2, 3\}$ atau {n: n adalah bilangan bulat antara 1 dan 3}

Himpunan bilangan bulat positif:

$\{1, 2, 3, 4, \ldots\}$ atau {n: n > 0}

Himpunan kosong:

{ } atau {x: x adalah persegi bulat} atau $\emptyset$

Jenis-jenis Himpunan

Himpunan selanjutnya dikategorikan ke dalam berbagai jenis, berdasarkan elemen atau jenis elemen. Berbagai jenis himpunan dalam teori himpunan dasar ini adalah:

Himpunan terbatas: Jumlah elemen terbatas.

Himpunan tak terbatas: Jumlah elemen tidak terbatas.

Himpunan kosong: Tidak memiliki elemen.

Himpunan singleton: Ini hanya memiliki satu elemen.

Himpunan yang sama: Dua set sama jika mereka memiliki elemen yang sama.

Himpunan setara: Dua set setara jika mereka memiliki jumlah elemen yang sama.

Himpunan Power: Satu set dari setiap subset yang mungkin.

Himpunan universal: Setiap himpunan yang berisi semua set yang sedang dipertimbangkan.

Subset: Ketika semua elemen himpunan A milik himpunan B, maka A adalah himpunan bagian dari B.

Teori Rumus Himpunan

$n(A \cup B) = n(A) + n(B) - n(A \cap B)$

$n(A \cup B) = n(A) + n(B)$ {ketika A dan B adalah **himpunan terputus-putus**}

$n(U) = n(A) + n(B) - n(A \cap B) + n((A \cup B) c)$

$n(A \cup B) = n(A - B) + n(B - A) + n(A \cap B)$

$n(A - B) = n(A \cap B) - n(B)$

$n(A - B) = n(A) - n(A \cap B)$

$n(Ac) = n(U) - n(A)$

$n(P \cup Q \cup R) = n(P) + n(Q) + n(R) - n(P \cap Q) - n(Q \cap R) - n(R \cap P) + n(P \cap Q \cap R)$

Operasi Himpunan

Operasi yang ditetapkan meliputi penyatuan, persimpangan, perbedaan dan komplemen (Kaplinski et al., 2015). Dalam pemrograman, operasi ini dapat diimplementasikan menggunakan struktur kumpulan data.

$A \cup B$ (union): Elemen yang berada di A atau B.

$A \cap B$ (persimpangan): Elemen yang ada di A dan B.

$A - B$ (perbedaan): Elemen yang berada di A tetapi tidak di B.

A' (komplemen): Elemen yang tidak ada di A.

Gabungan dua himpunan X dan Y sama dengan himpunan elemen yang ada dalam himpunan X, dalam himpunan Y, atau dalam himpunan X dan Y (Bourbaki, 2004). Operasi ini dapat direpresentasikan sebagai;

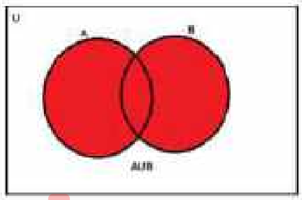
$X \cup Y = \{a: a \in X \text{ atau } a \in Y\}$

Mari kita perhatikan sebuah contoh, katakanlah; himpunan $A = \{1, 3, 5\}$ dan himpunan $B = \{1, 2, 4\}$ maka; $A \cup B = \{1, 2, 3, 4, 5\}$

Sekarang, mari kita belajar bagaimana kita bisa mewakili penyatuan dua set dalam **diagram Venn**.

Diagram Venn dari Himpunan Gabungan

Mari kita perhatikan himpunan universal U sedemikian rupa sehingga A dan B adalah himpunan bagian dari himpunan universal ini. Penyatuan dua himpunan A dan B didefinisikan sebagai himpunan semua elemen yang terletak pada himpunan A dan himpunan B atau kedua elemen dalam A dan B sama sekali (Roman, 2008). Penyatuan himpunan dilambangkan dengan simbol '∪'.



$C = A \cup B = \{x : x \in A \text{ atau } x \in B\}$

Rumus Jumlah Elemen dalam A gabung B (Hausdorff, 2021)

Pertimbangkan dua himpunan, A dan B, sehingga jumlah elemen dalam penyatuan A dan B dapat dihitung sebagai berikut.

$n(A \cup B) = n(A) + n(B) - n(A \cap B)$

Dimana:

$n(A \cup B)$ = Jumlah total elemen dalam A U B; disebut kardinalitas himpunan A U B

$n(A)$ = Jumlah elemen dalam A; disebut kardinalitas himpunan A

$n(B)$ = Jumlah elemen dalam B; disebut kardinalitas himpunan B

$n(A \cap B)$ = Jumlah elemen yang umum untuk A dan B; disebut kardinalitas himpunan $A \cap B$, yaitu irisan (*intersection*) B

Contoh-1:

Misalkan U adalah himpunan universal yang terdiri dari semua bilangan asli sampai 20 dan himpunan A dan B menjadi subset dari U yang didefinisikan sebagai A = {2, 5, 9, 15, 19} dan B = {8, 9, 10, 13, 15, 17}. Temukan $A \cup B$.

Penyelesaian:

Diketahui

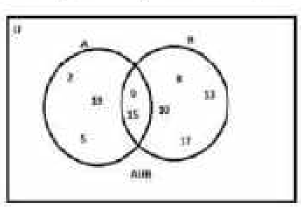
U = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20}

A = {2, 5, 9, 15, 19}

B = {8, 9, 10, 13, 15, 17}

$A \cup B$ = {2, 5, 8, 9, 10, 13, 15, 17, 19}

Ini dapat direpresentasikan menggunakan diagram Venn berikut:



Contoh-2:

Jika himpunan A berisi 13 elemen, himpunan B berisi 8 elemen dan perpotongan kedua himpunan ini berisi 5 elemen, kemudian cari jumlah elemen dalam A gabung B.

Penyelesaian:

Diketahui

Jumlah elemen dalam himpunan A = n (A) = 13

Jumlah elemen dalam himpunan B = n (B) = 8

Jumlah elemen dalam A persimpangan B = n (A ∩ B) = 5

Oleh karena itu,

$$
\begin{aligned}
n(A \cup B) &= n(A) + n(B) - n(A \cap B) \\
&= 13 + 8 - 5 \\
&= 21 - 5 \\
&= 16
\end{aligned}
$$

Oleh karena itu, jumlah elemen dalam A union B = n (A U B) = 16.

Gabungan dan Irisan dari Himpunan

Nomor Kardinal dari Himpunan

Jumlah elemen atau anggota yang berbeda dalam himpunan terbatas dikenal sebagai bilangan kardinal himpunan (Bagaria, 2014). Pada dasarnya, melalui kardinalitas, kita menentukan ukuran satu set. Nomor kardinal dari himpunan A dilambangkan sebagai n (A), dimana A adalah himpunan apa pun dan n (A) adalah jumlah anggota dalam himpunan A.

Pertimbangkan himpunan A yang terdiri dari bilangan prima kurang dari 10.

Set A ={2, 3, 5, 7}.

Karena himpunan A terdiri dari 4 elemen, oleh karena itu, jumlah kardinal himpunan A diberikan sebagai n (A) = 4.

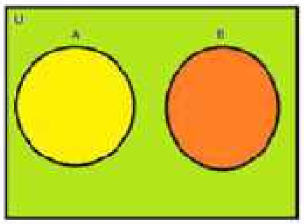
Properti yang terkait dengan perbedaan, gabungan, irisan, dan nomor kardinal dari suatu himpunan

i) Gabungan dari Himpunan Terputus-putus (Fraczak et al., 2013)

Jika A dan B adalah dua himpunan terbatas dan jika A ∩ B = ∅, maka

n (A ∪ B) = n (A) + n (B)

Dengan kata sederhana, jika A dan B adalah himpunan terbatas dan himpunan ini terputus-putus, maka jumlah kardinal Persatuan himpunan A dan B sama dengan jumlah jumlah kardinal himpunan A dan himpunan B.



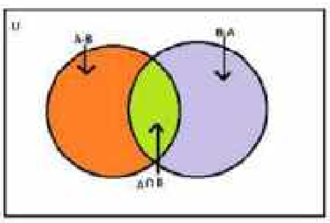
Penyatuan himpunan terputus-putus A dan B yang diwakili oleh diagram Venn diberikan oleh $A \cup B$ dan dapat dilihat bahwa $A \cap B = \emptyset$ karena tidak ada elemen yang umum untuk kedua himpunan.

ii) Gabungan Dua Himpunan (Erdös et al., 1982)

Jika A dan B adalah dua himpunan terbatas, maka

$$n(A \cup B) = n(A) + n(B) - n(A \cap B)$$

Sederhananya, jumlah elemen dalam penyatuan himpunan A dan B sama dengan jumlah bilangan kardinal himpunan A dan B, dikurangi irisan.



Pada gambar yang diberikan di atas daerah yang diarsir berbeda menggambarkan himpunan terputus-putus yang berbeda yaitu $A - B$, $B - A$ dan $A \cap B$ adalah tiga himpunan terputus-putus seperti yang ditunjukkan dan jumlah ini mewakili $A \cup B$. Oleh karena itu,

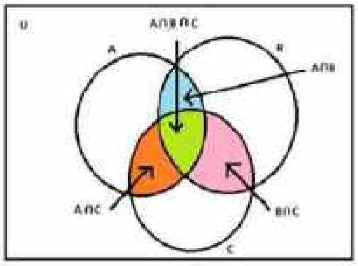
$$n(A \cup B) = n(A - B) + n(B - A) + n(A \cap B)$$

iii) Gabungan dari Tiga Himpunan (Pulaj, 2023)

Jika A, B, dan C adalah tiga himpunan hingga, maka;

$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(B \cap C) - n(A \cap C) + n(A \cap B \cap C)$$

Hal ini terlihat jelas dari diagram Venn bahwa penyatuan tiga himpunan akan menjadi jumlah kardinal himpunan A, himpunan B, himpunan C dan unsur-unsur umum dari tiga himpunan tidak termasuk unsur-unsur umum dari himpunan yang diambil berpasangan dua.



Contoh-1:

Total ada 200 siswa di kelas XI. 120 dari mereka belajar matematika, 50 siswa belajar perdagangan dan 30 siswa belajar matematika dan perdagangan. Temukan jumlah siswa yang

i) Belajar matematika tetapi tidak perdagangan

ii) Belajar perdagangan tetapi tidak matematika

iii) Belajar matematika atau perdagangan

Jawab:

Jumlah total siswa mewakili jumlah kardinal dari himpunan universal. Biarkan A menunjukkan himpunan siswa yang belajar matematika dan himpunan B mewakili siswa yang belajar perdagangan.

Jadi;

$n(U) = 200$
$n(A) = 120$
$n(B) = 50$
$n(A \cap B) = 30$

i) Disini, kita diminta untuk menemukan perbedaan himpunan A dan B.

$n(A) = n(A-B) = n(A) - n(A \cap B)$
$= 120 - 30$
$= 90$

Jumlah siswa yang belajar matematika tetapi tidak berdagang adalah 90.

ii) Demikian pula, disini, kita diminta untuk menemukan perbedaan himpunan B dan A

$n(B) = n(A-B) = n(B) - n(A \cap B)$
$= 50 - 30$
$= 20$

Jumlah siswa yang belajar perdagangan tetapi tidak matematika adalah 20.

iii) Jumlah siswa yang belajar matematika atau perdagangan

$n(A \cup B) \quad = n(A) + n(B) - n(A \cap B)$

$n(A \cup B) \quad = 120 + 50 - 30$

$\quad = 140$

Definisi dari Himpunan Komplemen

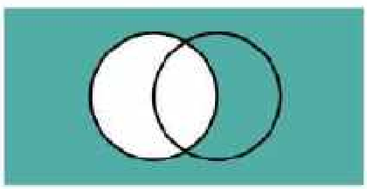
Jika U adalah himpunan universal dan A adalah himpunan bagian dari U maka komplemen A adalah himpunan dari semua anggota himpunan universal U yang bukan merupakan elemen A (Miller, 1984).

$$A' = x: x \in U \; and \; x \notin A$$

Atau, dapat dikatakan bahwa perbedaan himpunan universal U dan himpunan bagian A memberi kita pelengkap himpunan A.

Diagram Venn untuk himpunan komplemen

Diagram Venn untuk mewakili komplemen dari himpunan A diberikan oleh:



Contoh-1

Untuk membuatnya lebih jelas, pertimbangkan himpunan universal U dari semua bilangan asli kurang dari atau sama dengan 20 {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20}.

Misalkan himpunan A yang merupakan himpunan bagian dari U didefinisikan sebagai himpunan yang terdiri dari semua bilangan prima.

Dengan demikian kita dapat melihat bahwa A = {2, 3, 5, 7, 11, 13, 17, 19}

Sekarang komplemen dari himpunan A ini terdiri dari semua elemen yang ada dalam himpunan universal tetapi tidak dalam A. Oleh karena itu, A' diberikan oleh:

A'={1, 4, 6, 8, 9, 10, 12, 14, 15, 16, 18, 20}

Contoh-2

Biarkan U menjadi himpunan universal yang terdiri dari semua bilangan bulat lebih besar dari 5 tetapi kurang dari atau sama dengan 25. Biarkan A dan B menjadi himpunan bagian dari U yang didefinisikan sebagai:

B = {7, 9, 16, 18, 24}

Temukan komplemen set A dan B dan persimpangan kedua himpunan komplemen.

Penyelesaian: Himpunan universal didefinisikan sebagai:

U = {6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25}

Dimana, A = {9, 16, 25} dan

B = {7, 9, 16, 18, 24}

Himpunan komplemen A didefinisikan sebagai:

A' = {6, 7, 8, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 21, 22, 23, 24}

Demikian pula, himpunan komplemen B dapat diberikan oleh:

B' = {6, 8, 10, 11, 12, 13, 14, 15, 17, 19, 20, 21, 22, 23, 25}

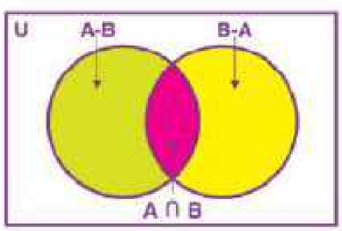
Irisan kedua himpunan komplemen diberikan oleh A'∩ B'.

A' ∩ B'= {6, 8, 10, 11, 12, 13, 14, 15, 17, 19, 20, 21, 22, 23}

Kita dapat melihat dari pembahasan di atas bahwa jika himpunan A adalah himpunan bagian dari himpunan universal U maka komplemen dari himpunan A, yaitu A' juga merupakan himpunan bagian dari U.

Perbedaan (selisih) dari Himpunan

Perbedaan dari dua himpunan A dan B adalah himpunan elemen yang ada dalam A tetapi tidak dalam B (Bruck, 1955). Hal ini dilambangkan sebagai A-B. Dalam diagram berikut, wilayah yang diarsir hijau mewakili perbedaan set A dan B (A-B). Dan wilayah yang diarsir kuning mewakili perbedaan B dan A (B-A).



Contoh

Misalkan A = {3, 4, 8, 9, 11, 12 } dan B = {1, 2, 3, 4, 5 }. Temukan A – B dan B – A.

Penyelesaian: Kita dapat mengatakan bahwa

A – B = {8, 9, 11, 12} karena elemen-elemen ini milik A tetapi tidak milik B

B – A = {1, 2, 5} karena elemen-elemen ini milik B tetapi tidak milik A.

Tupel

Tupel adalah urutan elemen yang tidak dapat diubah setelah dibuat. Dalam pemrograman, tupel sering digunakan untuk mengelompokkan data terkait.

Contoh

$(1, 'John', 25)$

Urutan (Sequence)

Urutan adalah urutan elemen yang dapat diakses menggunakan indeks. Dalam pemrograman, string, daftar, dan tupel adalah contoh urutan.

Contoh

- String: "Halo, Dunia"
- Daftar: [1, 2, 3, 4, 5]

Himpunan Daya (Power Sets)

Power set adalah himpunan semua himpunan bagian dari himpunan, termasuk himpunan kosong dan himpunan itu sendiri.

Contoh

$P\ (A)\ dari\ A = \{1,2\}\ adalah\ \{\{1\}, \{2\}, \{1,2\}\}$

Aplikasi di Bidang Informatika

Contoh sederhana penerapan konsep-konsep ini dalam manajemen basis data toko online. Implementasi dapat diperluas tergantung pada kebutuhan aplikasi dan sistem yang diimplementasikan.

Studi Kasus: Manajemen Database

Misalkan kita memiliki database pelanggan toko online:

```python
pelanggan = {'Alice', 'Bob', 'Charlie', 'David'}
pembelian_Alice = {'item1', 'item2', 'item3'}
pembelian_Bob = {'item2', 'item4', 'item5'}
```

Catatan: pelanggan (pelanggan)
pembelian (pembelian)

Aplikasi:

1. Atur operasi:
   - Agregat pelanggan yang melakukan pembelian: *customer_purchase= purchase_Alice ∪ purchase_Bob*
   - Pelanggan yang belum melakukan pembelian: *customer_not yet_purchase= pelanggan − customer_purchase*
2. Tupel dan urutan:
   Representasi data pembelian untuk setiap pelanggan sebagai tupel:
   (Alice, beli _Alice), (Bob, *beli* _Bob)
3. Power set:
   Gunakan power set untuk mengelola izin akses pengguna pada sistem.

Kesimpulan

Konsep dasar seperti himpunan, operasi himpunan, tupel, urutan, dan himpunan pangkat memegang peranan penting dalam dasar-dasar matematika yang diterapkan dalam dunia informatika. Memahami dan menerapkan konsep-konsep ini memberikan dasar yang kuat untuk pengembangan perangkat lunak, manajemen basis data, dan analisis data.

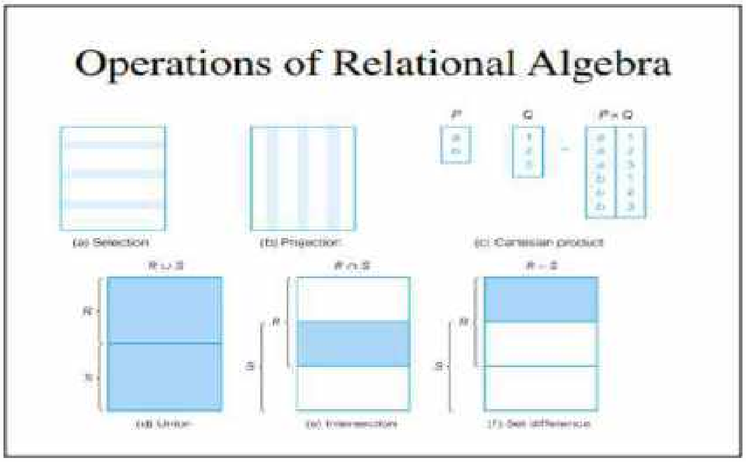
Himpunan dan operasi himpunan: Himpunan menyediakan cara logis untuk mengelompokkan dan mewakili data. Operasi Himpunan memungkinkan manipulasi data yang efisien, termasuk penyatuan, persimpangan, perbedaan, dan komplemen. Tupel dan Urutan: Tupel dan urutan (seperti string, daftar, dan tupel) menyediakan cara untuk mewakili dan mengelola data terstruktur. Urutan memfasilitasi akses data dan manipulasi dengan indeks.

Power Set: Power set menyediakan alat untuk mengelola struktur data yang kompleks dan analisis kombinatorial. Diterapkan dalam manajemen izin, analisis kompleksitas algoritma, dan representasi struktur data tingkat tinggi. Aplikasi di bidang informatika: Studi kasus tentang manajemen basis data toko online memberikan contoh konkret penerapan konsep-konsep ini. Memahami konsep-konsep ini membantu para profesional informatika dalam merancang solusi yang efisien dan efektif.

Penerapan konsep-konsep dasar ini membantu meningkatkan efisiensi dalam mengelola data, merancang sistem yang kuat, dan menangani masalah kompleks di dunia informatika. Dengan pemahaman yang kuat tentang dasar-dasar ini, para profesional dapat merancang solusi informatika yang kuat dan responsif terhadap tuntutan dunia digital yang terus berkembang.

15 KONSEP HUBUNGAN, KOMPOSISI RELASI, DAN SIFAT RELASI

Peta Konsep



(CSC 443)

Pendahuluan

Dalam dunia matematika dan ilmu komputer, konsep hubungan memainkan peran kunci dalam mewakili hubungan antara unsur-unsur himpunan. Hubungan tidak hanya membantu kita menggambarkan hubungan antara objek, tetapi juga membuka pintu untuk konsep lebih lanjut seperti komposisi hubungan dan sifat hubungan. Pemahaman mendalam tentang konsep-konsep ini memainkan peran integral dalam pemodelan dan analisis struktur data, hubungan antar entitas, dan pemecahan masalah yang melibatkan interaksi antar elemen.

Konsep Hubungan

Hubungan pada dasarnya adalah cara untuk menghubungkan elemen dari dua himpunan atau lebih. Dengan menggunakan pasangan terurut, relasi menyediakan cara yang fleksibel untuk mewakili hubungan antar objek. Misalnya, dalam konteks himpunan

A= {1, 2, 3} dan B= {a, b, c}, kita dapat memiliki relasi R= {(1, a), (2, b), (3, c)}, yang menyatakan bahwa elemen 1 dari A sesuai dengan elemen a dari B.

Komposisi Hubungan

Komposisi hubungan melibatkan menggabungkan dua atau lebih hubungan untuk membentuk hubungan baru. Ini menyediakan cara yang efisien untuk menggambarkan bagaimana hubungan antara elemen dari set yang berbeda dapat digabungkan. Dalam konteks komposisi hubungan, pemodelan hubungan antar entitas menjadi lebih luas dan dapat digunakan untuk menganalisis interaksi yang lebih kompleks.

Properti Hubungan

Properti relasi menyoroti properti tertentu yang dimiliki hubungan antar elemen. Refleksif, simetris, transitif dan antisimetris adalah beberapa sifat hubungan yang membantu dalam memahami sifat hubungan dan menerapkan konsep-konsep ini dalam berbagai konteks.

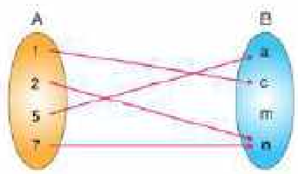
Dengan memahami konsep hubungan, komposisi hubungan, dan sifat hubungan, kita dapat menjelajahi dunia luas model matematika yang mewakili hubungan antar objek. Dalam konteks ilmu komputer, konsep ini adalah dasar untuk pemodelan struktur data, analisis algoritma, dan desain sistem yang melibatkan hubungan dan interaksi antar elemen. Dengan melangkah lebih jauh dalam pemahaman ini, kita dapat mengoptimalkan pemecahan masalah dan menangani kompleksitas di dunia yang terus berkembang.

Pembahasan

Konsep Hubungan

Dalam matematika dan ilmu komputer, hubungan adalah hubungan antara dua himpunan, yang dapat didefinisikan sebagai kumpulan pasangan terurut (Hotelling, 1992).

Himpunan pasangan terurut didefinisikan sebagai relasi (Contoh- 1):



Pemetaan ini menggambarkan relasi dari himpunan A ke himpunan B. Hubungan dari A ke B adalah himpunan bagian dari A x B. Pasangan yang dipesan adalah (1, c), (2, n), (5, a), (7, n). Untuk mendefinisikan relasi, kita menggunakan notasi di mana,

Himpunan {1, 2, 5, 7} mewakili **domain**.

Himpunan {a, c, n} mewakili **rentang**.

Contoh- 2:

Himpunan $A = \{1, 2, 3\}$ dan $B = \{a, b, c\}$ kemudian relasi R dari A ke B dapat didefinisikan sebagai kumpulan pasangan terurut, seperti $R = \{(1, a), (2, b), (3, c)\}$. Dengan kata lain, relasi menghubungkan elemen-elemen dari satu himpunan dengan elemen-elemen dari himpunan lain.

Hubungan Kosong

Hubungan kosong adalah hubungan dimana tidak ada hubungan antara unsur-unsur himpunan (Gasché, 2012). Misalnya, jika himpunan A = {1, 2, 3} maka, salah satu relasi kekosongan dapat berupa R = {x, y} di mana, |x − y| = 8. Untuk hubungan kosong,

$R = \varphi \subset A \times A$

Hubungan Universal

Sebuah universal (atau hubungan penuh) adalah jenis hubungan di mana setiap elemen dari himpunan terkait satu sama lain (Jónsson, 1956). Pertimbangkan himpunan A = {a, b, c}. Sekarang salah satu hubungan universal adalah R = {x, y} di mana, $|x - y| \geq 0$. Untuk hubungan universal,

$R = A \times A$

Hubungan Identitas

Dalam hubungan identitas, setiap elemen dari himpunan hanya terkait dengan dirinya sendiri (Deutsch & Garbacz, 2002). Misalnya, dalam himpunan A = {a, b, c}, relasi identitas akan menjadi I = {a, a}, {b, b}, {c, c}. Untuk hubungan identitas,

$I = \{(a, a), a \in A\}$

Hubungan terbalik (Invers)

Hubungan terbalik terlihat ketika suatu himpunan memiliki unsur-unsur yang merupakan pasangan terbalik dari himpunan lain (Milne & Bhatnagar, 1998). Misalnya jika himpunan A = {(a, b), (c, d)}, maka relasi terbalik akan menjadi R^{-1} = {(b, a), (d, c)}. Jadi, untuk hubungan terbalik,

$R^{-1} = \{(b, a): (a, b) \in R\}$

Contoh-1

Misalkan P = {2, 3, 4} dan Q = {2, 4, 8, 9, 15}. Jika kita mendefinisikan hubungan R dari P ke Q dengan (p, q) R jika p membagi q maka kita mendapatkan R = {(2, 2), (2, 4), (4, 4), (2, 8), (4, 8), (3, 9), (3, 15)}

R^{-1} adalah kebalikan dari relasi R, yaitu {(2, 2), (4, 2), (4, 4), (8, 2), (8, 4), (9, 3), (15, 3)}

Contoh-2

Jika A adalah matriks yang mewakili relasi R,

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

maka matriks yang mewakili R^{-1} hubungan, misalkan B, diperoleh dengan mentransposisi matriks A,

$$A^T = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Hubungan refleksif

Dalam hubungan refleksif, setiap elemen memetakan dirinya sendiri (Finlay, 2003). Misalnya, perhatikan himpunan A = {1, 2}. Sekarang contoh hubungan refleksif adalah R = {(1, 1), (2, 2), (1, 2), (2, 1)}. Hubungan refleksif adalah:

$(a, a) \in R$

Hubungan simetris

Dalam hubungan simetris, jika $a = b$ benar maka $b = a$ juga benar (Hewitt & Savage, 1955). Dengan kata lain, relasi R simetris hanya jika $(b, a) \in R$ benar ketika $(a, b) \in R$. Contoh relasi simetris adalah $R = \{(1, 2), (2, 1)\}$ untuk himpunan $A = \{1, 2\}$. Jadi, untuk hubungan simetris,

$a\,R\,b \Rightarrow b\,R\,a, \forall\, a, b \in A$

Contoh-2

Misalkan $A = \{1, 2, 3, 4\}$, dan relasi R didefinisikan pada himpunan A, maka relasi $R = \{(1, 1), (1, 2), (2, 1), (2, 2), (2, 4), (3, 2), (4, 2), (4, 4)\}$

(a) $\{(1, 2), (2, 1), (2, 4), (4, 2)\}$ simetris karena jika (a, b) adalah R maka (b, a) juga R. Disini (1, 2) dan (2, 1) R, serta (2, 4) dan (4, 2) R.

(b) Hubungan $R = \{(2, 3), (2, 4), (4, 2)\}$ tidak simetris karena (2, 3) R, tetapi (3, 2) R.

(c) Hubungan $R = \{(1, 1), (2,2), (4, 4)\}$ tidak simetris karena (1, 1) R dan 1 = 1 dan, (2, 2) R dan 2 = 2.

Hubungan transitif

Untuk hubungan transitif, jika $(x, y) \in R, (y, z) \in R$, maka $(x, z) \in R$ (Robinson, 1964). Untuk hubungan transitif,

$a\,R\,b$ dan $b\,R\,c \Rightarrow a\,R\,c\ \forall\, a, b, c \in A$

Hubungan Kesetaraan

Jika suatu hubungan refleksif, simetris dan transitif pada saat yang sama, itu dikenal sebagai hubungan kesetaraan (Tonneau, 2001).

LATIHAN

1. Misalkan $A = \{a, b, c\}$ dan $B = \{a, b, c, d\}$.

Hubungan $R_1 = \{(a, a), (b, b), (c, c)\}$
Hubungan $R_2 = \{(a, a), (a, b), (a, c), (a, d)\}$

Tentukanlah:
$R_1 \cap R_2 = \{(a, a)\}$

$R_1 \cup R_2 = \{(a, a), (b, b), (c, c), (a, b), (a, c), (a, d)\}$

$R_1 - R_2 = \{(b, b), (c, c)\}$

$R_2 - R_1 = \{(a, b), (a, c), (a, d)\}$

Komposisi Hubungan

Komposisi hubungan terjadi ketika kita memiliki dua atau lebih hubungan dan menggabungkannya untuk membentuk hubungan baru (Chan, 1998). Jika R adalah relasi dari A ke B dan S adalah relasi dari B ke C, maka komposisi relasi R dan S dilambangkan sebagai $R \circ S$ (Goguen, 1967). Komposisi ini menghasilkan hubungan baru yang menggambarkan bagaimana unsur-unsur A berhubungan dengan unsur-unsur C (Arbab, 2004).

Contoh-1

Jika kemudian $R = \{(1,2),(3,4)\}$ *and* $S = \{(2,5),(4,6)\}$ R ∘ S = $\{(1,5),(3,6)\}$

Contoh-2

Misalkan R = {(1, 2), (1, 6), (2, 4), (3, 4), (3, 6), (3, 8)} adalah relasi dari himpunan A= { 1, 2, 3} dengan himpunan B = {2, 4, 6, 8} dan S = {(2, u), (4, s), (4, t), (6, t), (8, u)} adalah relasi dari himpunan C= {2, 4, 6, 8} dengan himpunan D= {s, t, u}.

Maka komposisi hubungan R dan S adalah

S ∘ R = {(1, u), (1, t), (2, s), (2, t), (3, s), (3, t), (3, u)}

Properti Hubungan

Beberapa properti yang sering dipertimbangkan dalam hubungan melibatkan properti tertentu yang mereka miliki. Beberapa sifat hubungan umum meliputi:

Refleksif: Suatu hubungan dikatakan refleksif jika setiap elemen dalam himpunan terkait dengan dirinya sendiri.

Simetris: Suatu hubungan dikatakan simetris jika setiap kali a berhubungan dengan b, maka b juga berhubungan dengan a.

Transitif: Suatu hubungan dikatakan transitif jika setiap kali a terkait dengan b dan b terkait dengan c, maka a terkait dengan c.

Antisimetris: Suatu hubungan dikatakan antisimetris jika setiap kali a terkait dengan b dan b terkait dengan a, maka a harus sama dengan b.

Properti relationship ini berguna dalam menganalisis dan memahami hubungan antar elemen dalam suatu himpunan.

Dengan konsep hubungan, komposisi hubungan, dan pemahaman sifat hubungan, kita dapat memodelkan dan menganalisis hubungan antar objek dalam berbagai konteks matematika dan ilmu komputer.

Kesimpulan

Untuk memahami dan menerapkan konsep hubungan, komposisi hubungan, dan sifat hubungan, kita dapat menyimpulkan bahwa ketiga konsep ini memberikan landasan matematika yang kuat untuk mewakili dan menganalisis hubungan antar objek dalam berbagai konteks. Konsep hubungan: Hubungan menyediakan cara formal untuk menggambarkan hubungan antara unsur-unsur himpunan. Pasangan yang teratur dalam hubungan memfasilitasi representasi yang fleksibel dan kompleks dari hubungan antar objek.

Komposisi hubungan: Komposisi hubungan memungkinkan menggabungkan dua atau lebih hubungan untuk membentuk hubungan baru. Menyediakan cara yang efisien untuk menganalisis dan mewakili hubungan antara elemen himpunan yang berbeda. Properti hubungan: Properti hubungan (refleksif, simetris, transitif, antisimetris) memberikan wawasan tentang sifat-sifat hubungan antar elemen. Properti ini berguna dalam menganalisis karakteristik hubungan dan memahami aspek-aspek spesifik dari interaksi antar objek.

Penerapan konsep-konsep ini sangat luas, baik dalam matematika murni maupun dalam ilmu komputer. Dalam konteks ilmu komputer, pemahaman yang kuat tentang hubungan dan komposisi hubungan memberikan dasar yang kuat untuk desain basis data, analisis algoritma, dan pengembangan model struktur data yang kompleks. Sifat hubungan membantu dalam menerapkan konsep-konsep ini dengan bijak, memastikan konsistensi, dan memahami sifat dasar hubungan antar elemen. Secara keseluruhan, konsep-konsep ini membuka pintu untuk memodelkan dunia nyata dengan cara yang lebih abstrak dan menyediakan kerangka kerja yang kuat untuk memahami interaksi dan keterkaitan dalam berbagai situasi.

16 OPERASI FUNGSI, HUBUNGAN, DAN KEBALIKAN DARI KASUS

Pendahuluan

Dalam dunia matematika, konsep fungsi dan hubungan adalah dasar untuk pemodelan hubungan dan interaksi antar elemen dalam himpunan. Melalui operasi seperti penambahan dan komposisi, kita dapat menggabungkan atau memanipulasi fungsi-fungsi ini untuk membentuk model matematika yang lebih kompleks. Selain itu, pemahaman tentang fungsi terbalik dan hubungan terbalik memberikan wawasan tentang pembalikan atau konversi arah hubungan ini.

Konsep Fungsi dan Hubungan

Fungsi adalah aturan atau pemetaan yang menghubungkan setiap elemen himpunan dengan tepat satu elemen dalam himpunan lain. Relasi, pada dasarnya, adalah seperangkat pasangan terurut yang menggambarkan hubungan antara elemen-elemen himpunan. Dalam pemodelan matematika, fungsi dan relasi digunakan untuk merepresentasikan dan memahami berbagai fenomena dari berbagai disiplin ilmu.

Operasi Fungsi dan Hubungan

Operasi pada fungsi melibatkan memanipulasi atau menggabungkan fungsi untuk membentuk fungsi baru. Penambahan, perkalian, dan komposisi adalah operasi umum yang digunakan untuk memanipulasi fungsi-fungsi tersebut. Disisi lain, operasi pada hubungan, seperti gabungan dan irisan, membantu dalam memahami dan memanipulasi hubungan antar elemen.

Fungsi dan Hubungan Terbalik

Fungsi terbalik adalah konsep yang melibatkan pembalikan fungsi, yaitu menemukan fungsi yang mengubah output suatu fungsi menjadi input aslinya. Ini memiliki analogi dengan hubungan terbalik, di mana kita mencari hubungan yang mengubah arah hubungan antar elemen.

Studi Kasus

Sebagai ilustrasi, kita dapat mengeksplorasi kasus operasi dan kebalikan dari dua fungsi dan hubungan tertentu. Misalnya, bagaimana penambahan dua fungsi dapat menghasilkan fungsi baru, atau bagaimana kebalikan dari suatu hubungan dapat mengubah arah hubungan antar elemen. Studi kasus ini memberikan pemahaman konkret tentang penerapan operasi fungsi, hubungan, dan konsep terbalik dalam pemodelan matematika.

Dengan mengeksplorasi konsep ini, kita akan mendapatkan pemahaman yang lebih dalam tentang bagaimana matematika mewakili dan menganalisis hubungan antara elemen dan bagaimana operasi ini berkontribusi pada kompleksitas model matematika.

Pembahasan

Jenis fungsi

(i) Fungsi $f: X \rightarrow Y$ didefinisikan sebagai satu-satu (injektif), jika gambar elemen X yang berbeda di bawah f berbeda (Wassermann, 1977), yaitu,
$x_1, x_2 \in X, f(x_1) = f(x_2) \rightarrow x_1 = x_2$

(ii) Fungsi : $X \rightarrow Y$ dikatakan ke (surjektif), jika setiap elemen Y adalah gambar dari beberapa elemen X di bawah f, yaitu, untuk setiap $y \in Y$ ada elemen $x \in X$ sehingga $f(x) = y$ (Kubrusly, 2011).

(iii) Fungsi $f: X \rightarrow Y$ dikatakan satu-satu dan ke (bijektif), jika f adalah satu-satu dan ke (Griffiths et al., 1970).

Komposisi fungsi

(i) Biarkan $f: A \rightarrow B$ dan $g: B \rightarrow C$ menjadi dua fungsi. Kemudian, komposisi f dan g, dilambangkan dengan $g \circ f$, didefinisikan sebagai fungsi $g \circ f: A \rightarrow C$ diberikan oleh $g \circ f(x) = g(f(x)), \forall x \in A$ (Schwartz, 1969).

(ii) Jika $f: A \rightarrow B$ dan $g: B \rightarrow C$ adalah satu-satu, maka $g \circ f: A \rightarrow C$ juga satu-satu (Watanabe, 1985).

(iii) Jika $f: A \rightarrow B$ dan $g: B \rightarrow C$ adalah onto, maka $g \circ f: A \rightarrow C$ juga onto.
Namun, kebalikan dari hasil yang disebutkan di atas (ii) dan (iii) tidak perlu benar. Selain itu, kami memiliki hasil berikut dalam arah ini (Watanabe, 1985).

(iv) Misalkan $f: A \rightarrow B$ dan $g: B \rightarrow C$ menjadi fungsi yang diberikan sedemikian rupa sehingga $g \circ f$ adalah satu-satu. Maka f adalah satu-satu (Sabidussi, 1959).

(v) Misalkan $f: A \rightarrow B$ dan $g: B \rightarrow C$ menjadi fungsi yang diberikan sedemikian rupa sehingga $g \circ f$ adalah onto (André, 1989). Kemudian g adalah onto.

Fungsi Invertible

(i) Fungsi $f: X \rightarrow Y$ didefinisikan sebagai invertible, jika ada fungsi $g: Y \rightarrow X$ sedemikian rupa sehingga $g \circ f = I_X$ dan $f \circ g = I_Y$. Fungsi g disebut kebalikan dari f dan dilambangkan dengan f^{-1} (Royden, 1963).

(ii) Fungsi $f: X \rightarrow Y$ dapat dibalik jika dan hanya jika f adalah fungsi bijektif (Chandramowliswaran, 2021).

(iii) Jika $f: X \rightarrow Y$, $g: Y \rightarrow Z$ dan $h: Z \rightarrow S$ adalah fungsi, maka $h \circ (g \circ f) = (h \circ g) \circ f$ (hukum Bylinski, 1990).

(iv) Biarkan f: X → Y dan g: Y → Z menjadi dua fungsi yang dapat dibalik. Kemudian $g \circ f$ juga dapat dibalik dengan $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$ (Lorens, 1964).

Operasi Biner

(i) Operasi biner * pada himpunan A adalah fungsi *: A × A → A. Kami menunjukkan * (a, b) dengan a*b.

(ii) Operasi biner * pada himpunan X disebut komutatif, jika a * b = b * a untuk setiap a, b ∈ X .

(iii) Operasi biner *: A × A → A dikatakan asosiatif jika (a * b) * c = a * (b * c), untuk setiap a, b, c ∈ A.

(iv) Diberikan operasi biner *: A × A→ A, elemen e ∈ A, jika ada, disebut identitas untuk operasi *, jika a * e = a = e * a, ∀ a ∈ A.

(v) Diberikan operasi biner *: A × A → A, dengan elemen identitas e dalam A, elemen a ∈ A, dikatakan dapat dibalik sehubungan dengan operasi *, jika ada elemen b dalam A sedemikian rupa sehingga a * b = e = b * a dan b disebut kebalikan dari a dan dilambangkan dengan a^{-1}.

Contoh-1:

Misalkan fungsi f: **R** →**R** didefinisikan oleh $f(x) = 4x - 1$, $\forall x \in$ **R**. Kemudian, tunjukkan bahwa f adalah satu-satu.

Jawab:

Untuk dua elemen $x_1, x_2 \in$ **R** dimana $f(x_1) = f(x_2)$,

$$4x_1 - 1 = 4x_2 - 1$$

$$4x_1 = 4x_2,$$

$$x_1 = x_2$$

Oleh karena itu f adalah satu-satu.

Contoh-2:

Jika $f = \{(5, 2), (6, 3)\}$, $g = \{(2, 5), (3, 6)\}$, tentukan f o g.

Jawab:

$f \circ g = \{(2, 2), (3, 3)\}$

Contoh-3:

Misalkan $f: \mathbf{R} \rightarrow \mathbf{R}$ menjadi fungsi yang didefinisikan oleh $f(x) = 4x - 3\ \forall x \in \mathbf{R}$. Kemudian tulis f^{-1}.

Mengingat bahwa f (x) = 4x − 3 = y,

$$4x = y + 3$$

$$x = \frac{y+3}{4}$$

Maka $\quad f^{-1}(x) = \frac{x+3}{4}$ $f^{-1}(y) = \frac{y+3}{4}$

Contoh-4:

Jika A = {1, 2, 3} dan f, g adalah relasi yang sesuai dengan himpunan bagian A × A yang ditunjukkan terhadapnya, manakah dari *f*, *g* yang merupakan fungsi? Mengapa?

$$f = \{(1, 3), (2, 3), (3, 2)\}$$

$$g = \{(1, 2), (1, 3), (3, 1)\}$$

Jawab:

f adalah fungsi karena setiap elemen A di tempat pertama dalam pasangan terurut terkait dengan hanya satu elemen A di tempat kedua sementara *g* bukan fungsi karena 1 terkait dengan lebih dari satu elemen A, yaitu, 2 dan 3.

Contoh-5:

Jika A = {a, b, c, d} dan f = {a, b), (b, d), (c, a), (d, c)}, tunjukkan bahwa f adalah satu-satu dari A ke A. Temukan f^{-1}

Jawab:

f adalah satu-satu karena setiap elemen A ditugaskan ke elemen yang berbeda dari himpunan A. Juga, f adalah onto karena f (A) = A. Selain itu, $f^{-1} = \{(b, a), (d, b), (a, c), (c, d)\}$.

Operasi Fungsi

Dalam matematika, operasi pada fungsi melibatkan berbagai manipulasi atau kombinasi fungsi untuk membentuk fungsi baru. Tiga operasi utama pada fungsi adalah:

Penambahan: Jika f(x) dan g(x) adalah dua fungsi, maka fungsi yang dihasilkan dari penambahan h(x)= f(x) + g(x) didefinisikan sebagai h(x)= f(x) + g(x) untuk setiap nilai x dalam domain f dan g.

Perkalian: Jika f(x) dan g(x) adalah dua fungsi, maka fungsi produk h(x)= f(x)· g(x) didefinisikan sebagai h(x)= f(x)·g (x) untuk setiap nilai x dalam domain f dan g.

Komposisi: Jika f(x) dan g(x) adalah dua fungsi, maka fungsi hasil komposisi h(x)= f(g(x)) didefinisikan sebagai h(x)= f(g(x)), yang berarti kita menerapkan fungsi g(x) terlebih dahulu dan kemudian menerapkan hasilnya ke f(x).

Jumlah f + g, perbedaan (selisih) f - g, perkalian a.f, perkalian f.g, dan hasil bagi f/g masing-masing didefinisikan sebagai berikut:

$(f+g)(x) = f(x) + g(x)$

$(f-g)(x) = f(x) - g(x)$

$(af)(x) = a\,f(x)$

$(f.g)(x) = f(x)\,g(x)$

$(f/g)(x) = f(x)/g(x),\ g(x) \neq 0$

Contoh:

Jika $f(x) = x+2$ dan $g(x) = x+3$, maka tentukan:

a. $(f+g)(x)$
b. $(f-g)(x)$
c. $(f \cdot g)(x)$
d. $\left(\frac{f}{g}\right)(x)$

Jawab:

a. $(f+g)(x)$

$$\begin{aligned} &= f(x) + g(x) \\ &= (x+2) + (x+3) \\ &= x+2+x+3 \\ &= x+x+2+3 \\ &= 2x+5 \end{aligned}$$

b. $(f-g)(x)$

$$\begin{aligned} &= f(x)-g(x) \\ &= (x+2) - (x+3) \\ &= x+2-x-3 \\ &= x-x+2-3 \\ &= -1 \end{aligned}$$

c. $(f \cdot g)(x)$

$$\begin{aligned} &= f(x) \cdot g(x) \\ &= (x+2)(x+3) \\ &= x^2 + 3x + 2x + 6 \\ &= x^2 + 5x + 6 \end{aligned}$$

d.

$$\left(\frac{f}{g}\right)(x) = \frac{f(x)}{g(x)} = \frac{(x+2)}{(x+3)} = \frac{x+2}{x+3}$$

Operasi Hubungan

Operasi relasi melibatkan manipulasi hubungan antara elemen dalam satu himpunan. Dua operasi hubungan secara umum adalah:

Gabungan (Union): Jika R dan S adalah dua relasi dari himpunan A ke B, maka relasi gabungan $R \cup S$ didefinisikan sebagai $R \cup S = \{(a, b) \mid (a, b) \in R$ atau $(a, b) \in S\}$.

Irisan (Intersection): Jika R dan S adalah dua relasi dari himpunan A ke B, maka relasi perpotongan $R \cap S$ didefinisikan sebagai $R \cap S = \{(a, b) \mid (a, b) \in R$ dan $(a, b) \in S\}$.

Fungsi dan Hubungan Terbalik (relasi invers)

Fungsi Invers: Jika $f: A \rightarrow B$ adalah fungsi, maka fungsi invers $f^{-1}: B \rightarrow A$ didefinisikan sebagai $f^{-1}(y)=x$ jika $f(x)=y$. Fungsi invers hanya ada jika f adalah fungsi satu-ke-satu (injektif) dan surjektif.

Hubungan Invers: Jika R adalah relasi dari A ke B, maka relasi terbalik R^{-1} didefinisikan sebagai $R^{-1} = \{(b, a) \mid (a, b) \in R\}$. Hubungan terbalik mengubah arah hubungan antar elemen.

Contoh-1:

Jika $f(x) = x-1$, maka cari invers $(f^{-1}(x))$

$$
\begin{aligned}
f(x) &= x-1 \\
y &= x-1 \\
x-1 &= y \\
x &= y+1
\end{aligned}
$$

$f-1(x) = x+1$

Contoh-2:

Misalkan kita memiliki dua fungsi: $f(x) = 2x + 3$ dan $g(x) = x^2$. Operasi penambahan kedua fungsi ini dapat dinyatakan sebagai $h(x) = f(x) + g(x)$, yang menghasilkan fungsi $h(x) = 2x^2+2x+3$

Selain itu, misalkan R adalah relasi dari $A = \{1, 2, 3\}$ ke $B = \{a, b, c\}$ dengan $R = \{(1, a), (2, b), (3, c)\}$. Operasi irisan dengan relasi $S = \{(1, a), (4, b), (3, c)$ menghasilkan $R \cap S = \{(1, a), (3, c)\}$.

Dalam konteks terbalik, jika $f(x)=2x+3$, maka fungsi invers $f^{-1}(y) = \frac{y-3}{2}$

Untuk relasi R, relasi terbalik $R^{-1} = \{(a,1), (b,2), (c,3)\}$.

Kesimpulan

Dalam memahami dan menerapkan konsep fungsi, relasi, dan operasi terbalik dari suatu kasus, kita dapat menyimpulkan bahwa: Operasi fungsi dan relasi; Penambahan, perkalian, dan komposisi adalah operasi penting pada fungsi yang memungkinkan kita untuk menggabungkan atau memanipulasi fungsi. Operasi relasi, seperti penyatuan dan persimpangan, membantu memanipulasi dan menganalisis hubungan antara elemen-elemen himpunan.

Fungsi dan hubungan invers: Fungsi terbalik melibatkan menemukan fungsi yang membalikkan efek fungsi asli, sementara hubungan terbalik mengubah arah hubungan antar elemen. Fungsi invers hanya ada jika fungsi aslinya adalah fungsi satu-ke-satu (injektif) dan surjektif.

Aplikasi untuk studi kasus: Melalui studi kasus konkret, kita dapat melihat bagaimana operasi ini diterapkan dalam konteks matematika dan ilmu komputer. Contoh penerapan operasi fungsi dan relasi memberikan pemahaman nyata tentang kompleksitas pemodelan matematika. Signifikansi dalam pemodelan matematika: Konsep operasi dan invers memiliki signifikansi besar dalam pemodelan matematika, membantu kita memahami dan menganalisis interaksi kompleks antara objek. Dalam ilmu komputer, pemahaman ini diterapkan dalam desain algoritma, manajemen data, dan analisis struktur data.

Dengan memahami konsep-konsep ini, kita dapat lebih akurat mewakili dan memanipulasi hubungan matematika, membuka jalan bagi solusi yang lebih efisien dan kompleks dalam berbagai disiplin ilmu. Fungsi, hubungan dan operasi terbalik tidak hanya memperkaya pemahaman kita tentang matematika, tetapi juga memainkan peran kunci dalam pengembangan model matematika yang memadai dalam penelitian dan aplikasi dunia nyata.

logmat2

ORIGINALITY REPORT

12%	11%	4%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	noexperiencenecessarybook.com Internet Source	2%
2	docplayer.info Internet Source	1%
3	repository.akprind.ac.id Internet Source	1%
4	fr.scribd.com Internet Source	1%
5	adoc.pub Internet Source	1%
6	www.scribd.com Internet Source	1%
7	eprints.akprind.ac.id Internet Source	1%
8	fmipa.unsrat.ac.id Internet Source	<1%
9	idoc.pub Internet Source	<1%

10	pt.scribd.com Internet Source	<1 %
11	Submitted to Universitas 17 Agustus 1945 Surabaya Student Paper	<1 %
12	pt.slideshare.net Internet Source	<1 %
13	Submitted to Universitas Muhammadiyah Sinjai Student Paper	<1 %
14	wisnupratama0.wordpress.com Internet Source	<1 %
15	Submitted to Universitas Amikom Student Paper	<1 %
16	www.slideshare.net Internet Source	<1 %
17	Submitted to Forum Komunikasi Perpustakaan Perguruan Tinggi Kristen Indonesia (FKPPTKI) Student Paper	<1 %
18	123dok.com Internet Source	<1 %
19	Submitted to Universitas Diponegoro Student Paper	<1 %

20	David J. John. "Constructing an even grammar from a 2-element sample", Proceedings of the 43rd annual southeast regional conference on - ACM-SE 43 ACM-SE 43, 2005 Publication	<1 %
21	Anderson, Natalie M.. "The Influence of a Course on Assessment for Inservice Secondary Mathematics Teachers", Utah State University, 2022 Publication	<1 %
22	Submitted to Trident University International Student Paper	<1 %
23	Mozart W. Talakua. "TITIK-ANTARA DI DALAM RUANG METRIK DAN RUANG INTERVAL METRIK", BAREKENG: Jurnal Ilmu Matematika dan Terapan, 2007 Publication	<1 %
24	Francis Y. Rumlawang, Harimanus Batkunde. "SIFAT-SIFAT INTEGRAL RIEMANN-STIELTJES", BAREKENG: Jurnal Ilmu Matematika dan Terapan, 2007 Publication	<1 %
25	Reed, Logan. "The Pólya-Szegš‘ Conjecture on Polygons: A Numerical Approach.", Rutgers, The State University of New Jersey - Camden, 2023 Publication	<1 %

26	Aji Permana Putra. "ANALISIS KESULITAN BELAJAR MATEMATIKA PADA TOPIK LOGIKA DI SMK MUHAMMADIYAH 3 KLATEN UTARA", Academy of Education Journal, 2019 Publication	<1 %
27	Alfitrisni Yuliana Kolins, Wahyuningsih Wahyuningsih, Nurfitriah Safrudin, Muhamad Epi Rusdin. "Analisis Kesalahan Peserta Didik dalam Menyelesaikan Soal Matematika pada Fungsi Komposisi dan Fungsi Invers", AlphaMath : Journal of Mathematics Education, 2020 Publication	<1 %
28	Ratuanik Mesak. "DESAIN PEMBELAJARAN PADA MATERI HIMPUNAN MENGGUNAKAN MODEL PROBLEM BASED LEARNING", Asimtot : Jurnal Kependidikan Matematika, 2019 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

logmat2

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57

PAGE 58

PAGE 59

PAGE 60

PAGE 61

PAGE 62

PAGE 63

PAGE 64

PAGE 65

PAGE 66

PAGE 67

PAGE 68

PAGE 69

PAGE 70

PAGE 71

PAGE 72

PAGE 73

PAGE 74

PAGE 75

PAGE 76

PAGE 77

PAGE 78

PAGE 79

PAGE 80

PAGE 81

PAGE 82

PAGE 83

PAGE 84

PAGE 85

PAGE 86

PAGE 87

PAGE 88

PAGE 89

PAGE 90

PAGE 91

PAGE 92

PAGE 93

PAGE 94

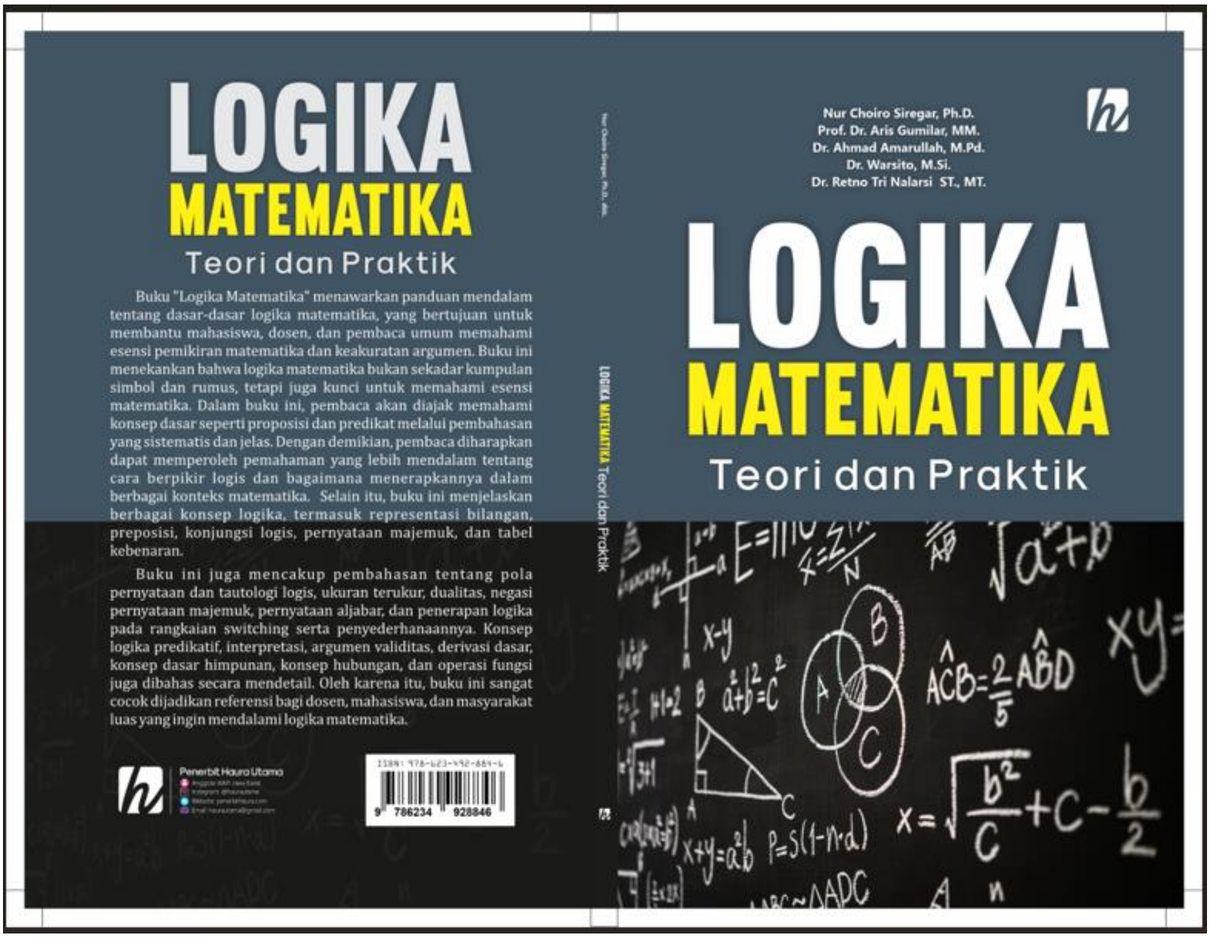
PAGE 95

PAGE 96

PAGE 97

PAGE 98

PAGE 99


LOGIKA
MATEMATIKA
Teori dan Praktik
Buku "Logika Matematika" menawarkan panduan mendalam tentang dasar-dasar logika matematika, yang bertujuan untuk membantu mahasiswa, dosen, dan pembaca umum memahami esensi pemikiran matematika dan keakuratan argumen. Buku ini menekankan bahwa logika matematika bukan sekadar kumpulan simbol dan rumus, tetapi juga kunci untuk memahami esensi matematika. Dalam buku ini, pembaca akan diajak memahami konsep dasar seperti proposisi dan predikat melalui pembahasan yang sistematis dan jelas. Dengan demikian, pembaca diharapkan dapat memperoleh pemahaman yang lebih mendalam tentang cara berpikir logis dan bagaimana menerapkannya dalam berbagai konteks matematika. Selain itu, buku ini menjelaskan berbagai konsep logika, termasuk representasi bilangan, preposisi, konjungsi logis, pernyataan majemuk, dan tabel kebenaran.
Buku ini juga mencakup pembahasan tentang pola pernyataan dan tautologi logis, ukuran terukur, dualitas, negasi pernyataan majemuk, pernyataan aljabar, dan penerapan logika pada rangkaian switching serta penyederhanaannya. Konsep logika predikatif, interpretasi, argumen validitas, derivasi dasar, konsep dasar himpunan, konsep hubungan, dan operasi fungsi juga dibahas secara mendetail. Oleh karena itu, buku ini sangat cocok dijadikan referensi bagi dosen, mahasiswa, dan masyarakat luas yang ingin mendalami logika matematika.
Penerbit Haura Utama
ISBN: 978-623-492-884-6
9 786234 928846
LOGIKA MATEMATIKA Teori dan Praktik
Nur Choiro Siregar, Ph.D.
Prof. Dr. Aris Gumilar, MM.
Dr. Ahmad Amarullah, M.Pd.
Dr. Warsito, M.Si.
Dr. Retno Tri Nalarsi ST., MT.
LOGIKA
MATEMATIKA
Teori dan Praktik

LOGIKA MATEMATIKA

Teori dan Praktik

Nur Choiro Siregar, Ph.D.
Prof. Dr. Aris Gumilar, MM.
Dr. Ahmad Amarullah, M.Pd.
Dr. Warsito, M.Si.
Dr. Retno Tri Nalarsi ST., MT.

Haura Utama

Logika Matematika: Teori dan Praktik,
karya Nur choiro Siregar, Ph.D., dkk ,
diterbitkan pertama kali oleh Penerbit Haura Utama, 2024

15 x 23 cm, 174 hlm

Editor dan Penata isi: Sri
Perancang sampul: Nita

CV. Haura Utama
Anggota IKAPI Nomor 375/JBA/2020
Nagrak, Benteng, Warudoyong, Sukabumi
+62877-8193-0045 haurautama@gmail.com

Cetakan I, Juni 2024

ISBN: 978-623-492-884-6

penerbithaura.com

Kata Pengantar

بسم الله الر حمن الر حيم

Assalamualaikum Warohmatullohi Wabarokatuh.

Dengan nama Allah Yang Maha Pemurah, Maha Penyayang. Pertama-tama, penulis ingin mengucapkan terima kasih kepada Allah karena telah memberi penulis kekuatan dan kesabaran sehingga ia dapat menyempurnakan buku teks berjudul **"Logika Matematika: Teori dan Praktik**".

Logika matematika adalah cabang ilmu yang membawa kita ke dalam pemahaman mendalam tentang pikiran manusia, keakuratan argumen, dan dasar-dasar pemikiran matematika. Buku ini disajikan sebagai panduan bagi mahasiswa, dosen, dan pembaca yang ingin mengetahui pengetahuan dasar logika matematika. Logika matematika bukan hanya kumpulan simbol dan rumus, tetapi juga merupakan kunci untuk memahami esensi matematika itu sendiri. Dalam buku ini, pembaca akan diajak untuk memahami konsep dasar seperti proposisi dan predikat. Melalui pembahasan yang sistematis dan jelas, diharapkan pembaca dapat memperoleh pemahaman yang lebih dalam tentang cara berpikir logis dan menerapkannya dalam berbagai konteks matematika.

Buku ini juga menjelaskan konsep logika, representasi bilangan, preposisi, konjungsi logis, pernyataan majemuk, dan tabel kebenaran, pola pernyataan dan tautologi logis, ukuran terukur, dualitas, negasi pernyataan majemuk, pernyataan aljabar, penerapan logika pada rangkaian switching, penyederhanaan, konsep logika predikatif, interpretasi, argumen validitas, derivasi dasar, konsep dasar himpunan, konsep hubungan, operasi fungsi. Oleh karena itu, buku ini sangat tepat untuk dijadikan referensi bagi dosen, mahasiswa, dan masyarakat luas.

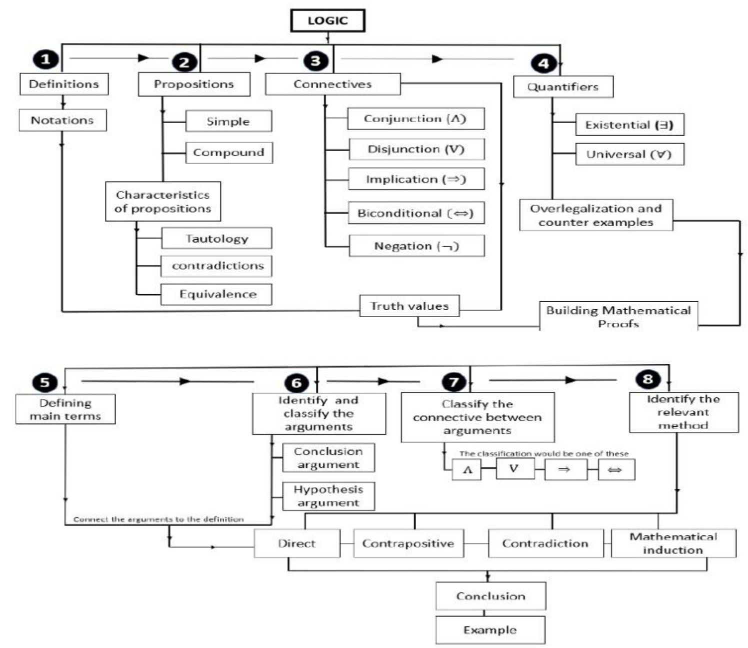
Semoga bermanfaat dan saya harap buku ini dapat membantu Anda menjelajahi keindahan logika matematika dengan sukacita dan kepuasan.

Nur Choiro Siregar

Daftar Isi

1
Pengantar, Pemahaman dan Konsep Logika

Peta Konsep



Gambar 1. Peta Konsep Logika dan Bukti Matematis (Issa & Majuma, 2019)

Pendahuluan

Matematika adalah ilmu pasti. Setiap pernyataan matematika harus tepat. Oleh karena itu, harus ada penalaran yang tepat dalam setiap bukti matematis. Penalaran yang tepat melibatkan logika. Belajar logika membantu dalam meningkatkan kemampuan seseorang dalam penalaran sistematis dan logis. Ini juga mengembangkan keterampilan memahami berbagai pernyataan dan validitasnya. Logika memiliki aplikasi skala luas dalam desain sirkuit, pemrograman komputer. Oleh karena itu, studi logika menjadi penting.

Pembahasan

Pernyataan dan nilai kebenarannya

Ada berbagai sarana komunikasi, yaitu lisan dan tulisan. Sebagian besar komunikasi melibatkan penggunaan bahasa di mana ide-ide disampaikan melalui kalimat.

Ada berbagai jenis kalimat seperti:

1. Deklaratif (Asertif)→ ringkas dan jelas
2. Imperatif (Perintah atau permintaan)
3. Seruan (Emosi, kegembiraan)
4. Interogatif (Pertanyaan)

Pernyataan

Pernyataan adalah kalimat deklaratif yang benar atau salah, tetapi tidak keduanya sekaligus. Pernyataan dilambangkan dengan huruf p, q, r, ...

Misalnya:

a. 3 adalah angka ganjil
b. 5 adalah persegi yang sempurna
c. Matahari terbit di timur
d. $x + 3 = 6$, ketika $x = 3$

Nilai Kebenaran

Sebuah pernyataan bisa Benar atau Salah. Nilai Kebenaran dari pernyataan "benar" didefinisikan sebagai T (*TRUE*) dan pernyataan "salah" didefinisikan sebagai F (*FALSE*)

Catatan: 0 dan 1 juga dapat digunakan untuk T dan F masing-masing

Pertimbangkan pernyataan berikut:

a. Tidak ada bilangan prima antara 23 dan 29
b. Matahari terbit dari barat
c. Kuadrat bilangan real adalah negatif
d. Jumlah sudut segitiga bidang adalah 180°

Di sini nilai kebenaran pernyataan 1 dan 4 adalah T dan nilai kebenaran pernyataan 2 dan 3 adalah F.

Catatan: Kalimat seperti seruan, interogatif (pertanyaan), imperatif (perintah/permintaan), tidak dianggap sebagai pernyataan karena nilai kebenaran pernyataan tidak dapat ditentukan.

Kalimat terbuka

Kalimat terbuka adalah kalimat yang kebenarannya dapat bervariasi sesuai dengan beberapa keadaan, yang tidak disebutkan dalam kalimat.

Catatan: Kalimat terbuka tidak dianggap sebagai pernyataan logis.

Misalnya:

a. $x \times 5 = 20$

Ini adalah kalimat terbuka karena kebenarannya tergantung pada nilai x (jika x = 4 benar, dan jika x ≠ 4 salah).

b. Makanan Cina sangat enak.

Ini adalah kalimat terbuka karena kebenaran bervariasi dari individu ke individu.

LATIHAN 1

Nyatakan yang mana dari kalimat berikut yang merupakan pernyataan. Jika ada pernyataan, tuliskan nilai kebenarannya.

1. Matahari adalah bintang.
2. Semoga Tuhan memberkati Anda!
3. Jumlah sudut interior segitiga dalam ruang Euclidean adalah 180°.
4. Setiap bilangan real adalah bilangan kompleks.
5. Mengapa Anda kesal?
6. Setiap persamaan kuadrat memiliki dua akar nyata.
7. $\sqrt{-9}$ adalah bilangan rasional.
8. $x^2 - 3x + 2 = 0$, dimana bahwa x = -1 atau x = -2.
9. Tolong ambilkan saya segelas air.
10. Dia orang yang baik.
11. Dua adalah satu-satunya bilangan prima genap.
12. Sungguh pemandangan yang mengerikan!
13. Jangan ganggu.

14. $x^2 - 3x - 4 = 0, x = -1$.
15. Bisakah Anda berbicara bahasa Prancis?
16. Kuadrat dari bilangan real apa pun adalah positif.
17. Warnanya merah.
18. Setiap jajaran genjang adalah belah ketupat.

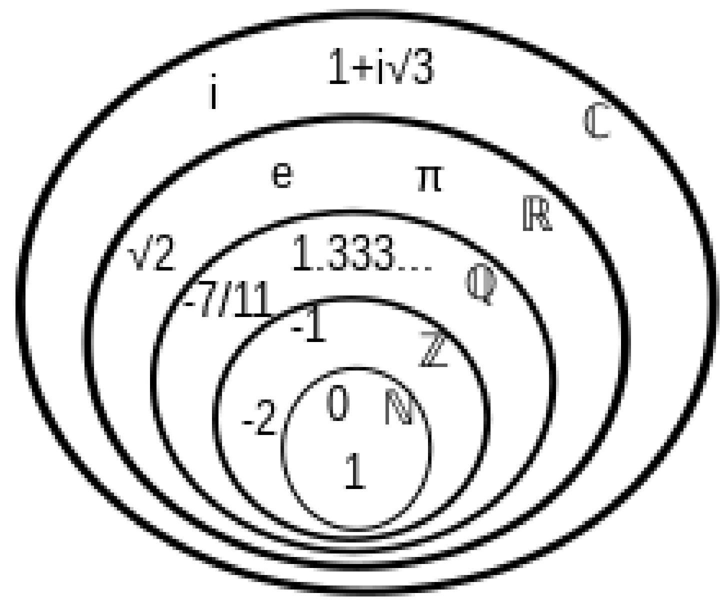
Penyelesaian:

1. Matahari adalah bintang. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
2. Semoga Tuhan memberkati Anda! (Ini adalah tanda seru; maka ini bukan pernyataan).
3. Jumlah sudut interior segitiga adalah 180°. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
4. Setiap bilangan real adalah bilangan kompleks. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
5. Mengapa Anda kesal? (Ini adalah kalimat interogatif; maka itu bukan pernyataan).
6. Setiap persamaan kuadrat memiliki dua akar nyata. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").
7. $\sqrt{-9}$ adalah bilangan rasional. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").
8. $x^2 - 3x + 2 = 0$, mengisyaratkan bahwa x = -1 atau x = -2. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").

9. Tolong ambilkan saya segelas air. (Ini adalah kalimat penting; karenanya ini bukan pernyataan).
10. Dia orang yang baik. (Ini adalah kalimat terbuka; maka ini bukan pernyataan).
11. Dua adalah satu-satunya bilangan prima genap. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
12. Sungguh pemandangan yang mengerikan! (Ini adalah tanda seru; maka ini bukan pernyataan).
13. Jangan ganggu. (Ini adalah kalimat penting; karenanya ini bukan pernyataan).
14. $x^2 - 3x - 4 = 0, x = -1$. (Ini adalah pernyataan yang benar, maka nilai kebenarannya adalah "T").
15. Bisakah Anda berbicara bahasa Prancis? (Ini adalah kalimat interogatif; maka ini bukan pernyataan).
16. Kuadrat dari bilangan real apa pun adalah positif. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F"); karena, 0 adalah bilangan real dan kuadrat dari 0 adalah 0 yang tidak positif atau negatif.
17. Warnanya merah. (Ini adalah kalimat terbuka; karenanya ini bukan pernyataan); kebenaran kalimat ini tergantung pada referensi ke kata ganti "itu".
18. Setiap jajaran genjang adalah belah ketupat. (Ini adalah pernyataan yang salah, maka nilai kebenarannya adalah "F").

2
Representasi Angka dan Aplikasi Dalam Algoritma Sederhana

Peta Konsep



Gambar 2.1 Wikipedia (2023)

N	Bilangan asli	Bilangan asli adalah angka 1, 2, 3, dll., Mungkin juga termasuk 0. Beberapa definisi, termasuk standar ISO 80000-2, memulai bilangan asli dengan 0, sesuai dengan bilangan bulat non-negatif 0, 1, 2, 3, ..., sedangkan definisi lain dimulai dengan 1.	0, 1, 2, 3, 4, 5, ... atau 1, 2, 3, 4, 5, ... N_0 dan N_1 terkadang digunakan
Z	Bilangan bulat	Bilangan bulat adalah nol (0), bilangan asli positif, atau	..., −5, −4, −3, −2, −1,

tergantung pada persyaratan aplikasi. Pemahaman yang baik tentang representasi angka memungkinkan perancang algoritma untuk menerapkan langkah-langkah yang benar dan menghasilkan hasil yang diinginkan.

Pembahasan

Bilangan real

Sistem bilangan desimal

Sistem bilangan desimal adalah sistem bilangan yang paling umum digunakan di dunia sehari-hari. Dalam sistem ini, kita menggunakan sepuluh angka, yaitu 0 hingga 9, untuk mewakili nilai numerik. Nilai setiap posisi angka dalam angka desimal terkait dengan pangkat 10. Misalnya, dalam angka desimal, 1234 memiliki nilai berikut:

$(1 \times 10^3) + (2 \times 10^2) + (3 \times 10^1) + (4 \times 10^0)$

Sistem bilangan desimal memiliki banyak aplikasi dalam algoritma sederhana. Salah satu aplikasinya adalah algoritma tambahan (Jang et al., 2019). Ketika kita ingin menambahkan dua angka atau lebih, kita menggunakan representasi desimal untuk angka-angka itu, dan algoritma penjumlahan sederhana akan menghitung hasil penjumlahan dalam sistem angka desimal.

Selain itu, angka desimal juga digunakan dalam algoritma penyortiran. Algoritma sederhana seperti pengurutan gelembung atau penyortiran pemilihan dapat digunakan untuk mengurutkan daftar angka desimal, naik atau turun/mengurutkan data dari kecil ke besar dan turun mengurutkan data dari besar ke kecil (Skliarova et al., 2019). Ini adalah implementasi penting karena penyortiran adalah operasi yang sering digunakan dalam banyak jenis aplikasi.

Dalam dunia pemrograman komputer, angka desimal juga digunakan dalam algoritma konversi (Rhyne, 1970). Ketika kita perlu mengonversi angka dari satu sistem angka ke sistem angka lainnya, seperti dari desimal ke biner atau sebaliknya, algoritma konversi dapat digunakan untuk melakukan operasi dengan benar.

$312{,}45 = 3 \times 10^2 + 1 \times 10^1 + 2 \times 10^0 + 4 \times 10^{-1} + 5 \times 10^{-2}$

(3 ratusan, 1 puluhan, 2 satu, 4 persepuluh, 5 per seratus)

Sistem desimal: Basis = 10, angkanya adalah (0, 1,..., 9)

Representasi standar: ± 312,45

±	3 1 2,	4 5
Tanda Angka	Bilangan bulat/bagian bulat	Bagian desimal

Representasi Nomor Float yang Dinormalisasi

Representasi bilangan float yang dinormalisasi adalah cara untuk menggambarkan bilangan real di komputer (Boehm, 2020). Dalam representasi ini, bilangan float terdiri dari tiga komponen utama: Tanda (positif atau negatif), eksponen, dan mantissa (pecahan atau bagian desimal). Representasi float yang dinormalisasi memastikan bahwa mantissa memiliki digit pertama yang selalu 1, sehingga mengoptimalkan presisi dan rentang nilai yang dapat diwakili oleh angka float.

Penerapan representasi bilangan float yang dinormalisasi dalam algoritma sederhana adalah dalam perhitungan matematis yang melibatkan bilangan real. Misalnya, algoritma sederhana seperti menghitung akar kuadrat atau membagi dua bilangan real menggunakan representasi float yang

Definisi Error-The Actual Error

Kesalahan sebenarnya adalah perbedaan antara hasil yang dihasilkan oleh suatu algoritma atau percobaan dan nilai yang dianggap sebagai "nilai sebenarnya" atau solusi yang benar. Kesalahan dapat timbul dari berbagai sumber, seperti kesalahan pengukuran, pembulatan, aproksimasi, atau kesalahan dalam algoritma itu sendiri (Ralston & Rabinowitz, 2001). Dalam analisis numerik dan ilmu lainnya, memahami kesalahan dan kemampuan untuk mengukurnya adalah aspek penting dalam memastikan akurasi dan keandalan hasil.

Kesalahan sejati adalah ukuran sejauh mana hasil yang diberikan oleh suatu algoritma atau eksperimen mendekati nilai yang dianggap benar atau solusi yang benar. Kesalahan sebenarnya adalah perbedaan antara hasil algoritma dan nilai yang benar atau solusi yang benar, yang seringkali sulit atau bahkan tidak mungkin untuk diketahui dengan pasti. Oleh karena itu, kesalahan sebenarnya adalah perkiraan sejauh mana hasil kami dapat diandalkan dan sesuai dengan tujuan yang diinginkan.

Aplikasi konsep kesalahan dan kesalahan aktual dalam algoritma sederhana dapat ditemukan dalam berbagai konteks. Dalam perhitungan numerik, kesalahan sering diukur untuk mengukur keakuratan hasil perhitungan. Misalnya, dalam algoritma sederhana untuk menghitung akar kuadrat, kesalahan dapat dihitung dengan membandingkan hasil algoritma dengan nilai akar kuadrat yang diketahui. Semakin kecil kesalahan, semakin dekat hasil algoritma ke nilai yang benar.

Selain itu, dalam ilmu data, konsep kesalahan digunakan untuk mengevaluasi model statistik. Model-model ini menghasilkan prediksi atau perkiraan, dan kesalahan

digunakan untuk mengukur sejauh mana prediksi mendekati nilai aktual. Dalam hal ini, memahami kesalahan membantu kita memahami kualitas model dan mengidentifikasi area di mana model perlu ditingkatkan.

Kesalahan dapat dihitung jika nilai sebenarnya dapat dihitung:

Kesalahan Absolut = $E_t = |True\ value - Approximate\ value|$

Kesalahan Relatif (dalam persen)

$$\varepsilon_t = \left|\frac{True\ value - Approximate\ value}{True\ value}\right| 100\%$$

Definisi Error–Estimated Error

Estimasi kesalahan adalah upaya untuk mengukur atau memperkirakan besarnya kesalahan dengan tujuan untuk memahami sejauh mana hasil yang diperoleh dapat diandalkan.

Penerapan konsep kesalahan estimasi dalam algoritma sederhana dapat ditemukan dalam berbagai situasi. Misalnya, ketika kita melakukan perhitungan matematika sederhana seperti pembagian dengan angka yang memiliki banyak digit desimal, seringkali kita harus membulatkan hasil perhitungan agar lebih mudah dipahami. Dalam hal ini, kita tahu bahwa ada kesalahan karena pembulatan, dan kita dapat memperkirakan seberapa besar kesalahan tersebut.

Dalam pengolahan data dan statistik, mengukur kesalahan estimasi penting dalam mengukur akurasi estimasi parameter statistik. Misalnya, jika kita ingin memperkirakan rata-rata populasi berdasarkan sampel, kita akan menghitung rata-rata sampel dan juga mengukur seberapa besar kesalahan standar perkiraan. Perkiraan kesalahan dalam kasus ini membantu kami memahami sejauh mana perkiraan kami dapat dipercaya sebagai representasi dari populasi sebenarnya.

Selain itu, dalam pemodelan numerik seperti mendekati fungsi matematika kompleks dengan polinomial sederhana, kita sering menggunakan kesalahan yang diperkirakan untuk memahami sejauh mana perkiraan kita mendekati fungsi sebenarnya. Ini membantu kami mengevaluasi seberapa baik model kami mewakili fenomena yang ingin kami akses.

Perkiraan kesalahan digunakan ketika nilai sebenarnya tidak diketahui:

Estimasi Kesalahan Absolut:

$$E_a = |Present\ value - Previous\ value|$$

Estimasi Kesalahan Relatif (dalam persen):

$$\varepsilon_t = \left|\frac{Present\ value - Previous\ value}{Present\ value}\right| 100\%$$

Notasi

Hasil estimasi dikatakan benar hingga n digit desimal jika:

$$|Error| \leq 10^{-n}$$

Hasil estimasi dikatakan benar hingga n digit desimal pembulatan jika:

$$|Error| \leq \frac{1}{2} x\ 10^{-n}$$

Kesimpulan

Representasi angka adalah cara kita menggambarkan nilai numerik dalam bentuk yang dapat dipahami dan digunakan oleh komputer atau manusia. Ada berbagai sistem representasi angka, termasuk desimal, biner, oktal, dan heksadesimal yang digunakan dalam berbagai aplikasi dan algoritma. Penerapan representasi angka dalam algoritma sederhana sangat penting. Algoritma adalah serangkaian langkah yang diperlukan untuk menyelesaikan tugas atau masalah. Dalam banyak kasus, algoritma memerlukan manipulasi angka, seperti penambahan, pengurangan, perkalian, atau pembagian. Oleh karena itu, pemahaman yang baik tentang representasi angka diperlukan untuk merancang dan mengimplementasikan algoritma sederhana dengan benar.

Representasi angka desimal adalah yang paling umum digunakan dalam kehidupan sehari-hari dan dalam aplikasi seperti kalkulator. Banyak algoritma sederhana mengambil angka desimal sebagai input dan menghasilkan output dalam representasi desimal juga. Selain representasi desimal, algoritma sederhana juga dapat menggunakan representasi bilangan biner. Representasi biner sangat penting dalam dunia komputer karena komputer bekerja dengan kode biner. Banyak algoritma sederhana beroperasi pada data biner, seperti menyortir dan mencari algoritma

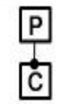
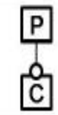
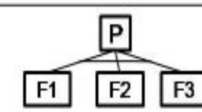
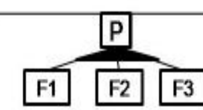
Angka yang memiliki representasi terbatas dalam satu sistem bilangan mungkin memiliki representasi tak terbatas dalam sistem bilangan lain. Dalam representasi floating-point yang dinormalisasi, bilangan real diwakili oleh tanda, eksponen, dan mantissa, di mana mantissa selalu memiliki digit pertama sama dengan 1. Hal ini memungkinkan representasi bilangan real yang efisien dan presisi tinggi. Penerapannya dalam algoritma sederhana terutama berfokus pada perhitungan matematis yang melibatkan bilangan real, seperti penambahan, pengurangan, perkalian dan pembagian. Representasi ini membantu algoritma sederhana dalam menghasilkan hasil yang mendekati nilai sebenarnya dengan tingkat presisi yang tinggi.

Dengan memahami representasi floating-point yang dinormalisasi, perancang algoritma dapat mengoptimalkan perhitungan numerik dalam berbagai aplikasi. Meskipun representasi ini mungkin lebih kompleks daripada bilangan bulat, penggunaannya dapat meningkatkan akurasi dan presisi dalam algoritma sederhana, memastikan bahwa hasil perhitungan mendekati nilai sebenarnya dengan tingkat kesalahan minimal. Pembulatan dan pemotongan membuatnya efisien untuk mewakili jumlah yang sangat kecil atau sangat besar. Representasi kesalahan tergantung pada bit yang digunakan.

3
Konsep Dasar Preposisi Dan Preposisi Dalam Algoritma Pemrograman

Peta Konsep

Tabel 3.1 Peta konsep logika proposisional (Lopez-Herrejon & Egyed, 2010)

Name	Diagram Notation	Propositional Logic
Mandatory		$P \Leftrightarrow C$
Optional		$C \Rightarrow P$
Exclusive-Or		$(F1 \Leftrightarrow (\neg F2 \wedge \neg F3 \wedge P)) \wedge$ $(F2 \Leftrightarrow (\neg F1 \wedge \neg F3 \wedge P)) \wedge$ $(F3 \Leftrightarrow (\neg F1 \wedge \neg F2 \wedge P))$
Inclusive-Or		$P \Leftrightarrow F1 \vee F2 \vee F3$
Requires	Cross feature arrow	$A \Rightarrow B$
Excludes	Cross feature arrow	$A \Rightarrow \neg B \equiv \neg(A \wedge B)$

Pendahuluan

Logika proposisional dan konsep preposisi memainkan peran penting dalam bidang algoritma pemrograman, menjembatani kesenjangan antara dunia abstrak logika matematika dan ranah praktis ilmu komputer (Sack, 2019).

Kedua subjek yang saling berhubungan ini memberikan kerangka dasar untuk penalaran tentang kebenaran, nilai-nilai pernyataan, dan hubungan antara berbagai elemen dalam algoritma. Dalam pengantar ini, kita akan mengeksplorasi pentingnya logika proposisional dan bagaimana preposisi menemukan aplikasi dalam algoritma pemrograman.

Logika Proposisional

Logika proposisional, sering disebut sebagai logika sentential atau kalkulus proposisional, berfungsi sebagai tulang punggung logis ilmu komputer dan membentuk dasar untuk penalaran logis dalam algoritma (Chowdhary, 2020). Pada intinya, logika proposisional berkaitan dengan proposisi, yang merupakan pernyataan yang bisa benar atau salah. Ini menawarkan satu set penghubung logis, seperti AND, OR, dan NOT, yang memungkinkan programmer untuk memanipulasi proposisi, memungkinkan mereka untuk membuat keputusan berdasarkan informasi dalam algoritma. Dengan menggunakan tabel kebenaran dan aturan formal inferensi, logika proposisional memastikan bahwa programmer dapat bernalar secara logis, yang sangat penting untuk desain algoritma, kebenaran, dan optimasi.

Preposisi sebagai Konstruksi Fundamental

Dalam konteks algoritma pemrograman, preposisi bukanlah konektor linguistik yang ditemukan dalam bahasa alami melainkan konstruksi mendasar yang digunakan untuk mengekspresikan hubungan dan kondisi antara elemen program yang berbeda (Chowdhary & Chowdhary, 2020). Preposisi membantu programmer menentukan aturan dan batasan yang memandu aliran suatu algoritma. Misalnya, "jika" "sementara" dan "sampai" adalah preposisi yang digunakan untuk menentukan kapan tindakan spesifik harus

diambil berdasarkan kondisi tertentu. Preposisi ini menentukan logika aliran kontrol dalam algoritma, memastikan bahwa eksekusi kode terjadi dengan cara yang terkontrol dan bermakna.

Pengambilan Keputusan yang Logis

Integrasi logika proposisional dan preposisi memungkinkan programmer untuk membuat keputusan yang tepat dan tepat dalam algoritma mereka (Baden et al., 2022). Mereka dapat menggunakan penghubung logis untuk menggabungkan proposisi dan mengekspresikan kondisi kompleks yang menentukan bagaimana suatu algoritma berperilaku. Pernyataan kondisional, loop, dan struktur percabangan dalam pemrograman sangat bergantung pada konstruksi logis ini. Misalnya, algoritma mungkin menyertakan pernyataan kondisional yang mengeksekusi blok kode tertentu hanya jika proposisi tertentu berlaku. Pengambilan keputusan logis ini adalah inti dari penulisan algoritma yang efisien, efektif, dan andal.

Penanganan dan Validasi Kesalahan

Preposisi dan logika proposisional juga sangat berharga untuk penanganan kesalahan dan validasi data dalam algoritma (Bordel et al., 2021). Dengan merumuskan preposisi yang memeriksa kondisi kesalahan tertentu atau memvalidasi data input, programmer dapat memastikan integritas dan keandalan algoritma mereka. Misalnya, algoritma yang memproses input pengguna dapat menggunakan logika proposisional untuk memverifikasi apakah input mematuhi kriteria tertentu, mencegah kesalahan yang tidak diinginkan atau kerentanan keamanan.

Pembahasan

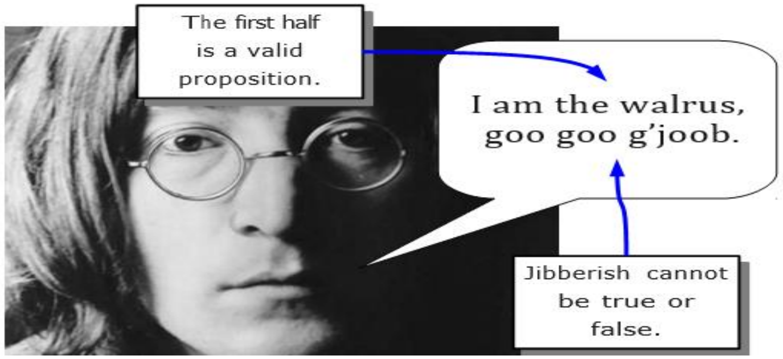
Logika Proposisional

Proposisi adalah pernyataan yang, dengan sendirinya, benar atau salah (Dewey, 1941).

- Anak anjing lebih manis dari anak kucing.
- Anak kucing lebih manis dari anak anjing.
- Bolt dapat berlari lebih cepat dari semua orang di ruangan ini.
- CS103 berguna untuk pesta koktail.
- Ini adalah entri terakhir dalam daftar ini.
- Saya seorang wanita lajang.
- Tempat ini akan meledak.
- Pesta besar ada di rumah malam ini.
- Kita bisa menari jika kita mau.
- Kita bisa meninggalkan teman-temanmu.

Hal-hal yang Bukan Proposisi

Hal-hal yang Bukan Proposisi



- Logika proposisional adalah sistem matematika untuk penalaran tentang proposisi dan bagaimana mereka berhubungan satu sama lain.
- Logika proposisional memungkinkan kita secara formal menyandikan bagaimana kebenaran berbagai proposisi mempengaruhi kebenaran proposisi lain.

- Tentukan apakah kombinasi proposisi tertentu selalu, kadang-kadang, atau tidak pernah benar.
- Tentukan apakah kombinasi proposisi tertentu secara logis memerlukan kombinasi lain.

Variabel dan Penghubung

- Logika proposisional adalah sistem matematika formal yang sintaksisnya ditentukan secara kaku (Braüner & Ghilardi, 2007).
- Setiap pernyataan dalam logika proposisional terdiri dari variabel proposisional yang digabungkan melalui penghubung logis (Allan, 2023).
- Setiap variabel mewakili beberapa proposisi, seperti "Anda menginginkannya" atau "Anda seharusnya memasang cincin di atasnya."
- Penghubung menyandikan bagaimana proposisi terkait, seperti "Jika Anda menginginkannya, Anda seharusnya memasang cincin di atasnya." (Bloom et al., 1980).

Variabel Proposisional

- Setiap proposisi akan diwakili oleh variabel proposisional.
- Variabel proposisional biasanya direpresentasikan sebagai huruf kecil, seperti p, q, r, s, dll.
- Jika kita membutuhkan lebih banyak, kita dapat menggunakan subskrip: p1, p2, dll.
- Setiap variabel dapat mengambil salah satu dari dua nilai: true atau false.

Pernyataan Bersyarat:

```python
if (x > 5) and (y < 10):
    # Perform an action if both x is greater than 5 and
```

Dalam contoh ini, operator dan digunakan untuk menggabungkan dua preposisi. Blok kode akan mengeksekusi hanya jika kedua preposisi (x > 5) dan (y < 10) benar.

Kontrol Loop:

```javascript
if (inputValue < 0):
    print("Error: Input value must be non-negative.")
```

Di sini, preposisi 'while' mengontrol loop, memastikan bahwa ia terus mengeksekusi hingga kondisi (userInput != "exit") menjadi false.

Penanganan Kesalahan:

```javascript
if (inputValue < 0):
    print("Error: Input value must be non-negative.")
```

Dalam hal ini, preposisi (inputValue < 0) digunakan untuk memvalidasi data input. Jika kondisi ini benar, pesan galat akan ditampilkan.

Percabangan logis

```c
if (temperature > 30) {
    // Code for a hot day.
} else if (temperature > 20) {
    // Code for a warm day.
} else {
    // Code for a cool day.
}
```

Contoh ini menunjukkan bagaimana preposisi digunakan untuk membuat cabang logis. Tergantung pada nilai variabel "suhu", blok kode yang berbeda akan dijalankan.

Fungsi Boolean

```python
def is_even(number):
    return (number % 2 == 0)

if is_even(8):
    print("The number is even.")
```

Dalam fungsi ini, preposisi '(angka% 2 == 0)' menentukan apakah angka yang diberikan genap. Fungsi mengembalikan nilai Boolean, dan pernyataan 'if' menggunakan hasilnya untuk membuat keputusan.

Contoh-contoh ini menunjukkan bagaimana logika proposisional dan preposisi sangat penting untuk algoritma pemrograman. Mereka memungkinkan programmer untuk menciptakan kondisi, loop, dan struktur pengambilan keputusan yang mengontrol aliran dan perilaku algoritma, memastikan bahwa kode dijalankan secara logis dan akurat sebagai respons terhadap input dan kondisi yang berbeda.

(a) FOL berkaitan dengan penalaran deduktif yang mengaktifkan penggunaan 'penghubung proposisional' seperti dan, atau, jika, tidak, dan pada penggunaan 'quantiers' seperti setiap, beberapa, tidak. Tetapi dalam bahasa biasa (termasuk bahasa biasa matematika informal) operator logis ini bekerja dengan cara yang sangat kompleks, memperkenalkan jenis ketidakjelasan dan kemungkinan ambiguitas yang tentu ingin kita hindari dalam argumen yang transparan secara logis. Apa yang harus dilakukan sejak zaman Aristoteles, para ahli logika telah menggunakan strategi 'membagi dan menaklukkan' yang melibatkan pengenalan bahasa yang disederhanakan dan

disiplin ketat. Bagi Aristoteles, bahasanya yang teratur adalah sebuah fragmen dari bahasa Yunani yang sangat kaku; Bagi kami, bahasa kami yang teratur sepenuhnya merupakan konstruksi formal artisial. Tapi bagaimanapun juga, rencananya adalah kita mengatasi bentangan penalaran dengan merumuskan ulang dalam bahasa yang sesuai dengan operator logis yang jauh lebih rapi, dan kemudian kita dapat mengevaluasi penalaran setelah disusun kembali ke dalam bentuk yang berperilaku lebih baik ini. Dengan cara ini, kita memiliki pembagian kerja. Pertama, kami mengklarifikasi struktur yang dimaksudkan dari argumen asli dengan menerjemahkannya ke dalam bahasa yang disederhanakan / diformalkan yang tidak ambigu. Kedua, ada urusan terpisah untuk menilai validitas argumen resimen yang dihasilkan.

Kami akan menggunakan bahasa formal yang sesuai yang berisi, khususnya, pengganti disiplin rapi untuk penghubung proposisional dan, atau, jika, tidak (dilambangkan secara standar $\wedge$ $\vee$ $\rightarrow$, ~) ditambah pengganti untuk quantifier bahasa biasa (kira-kira, menggunakan x untuk setiap x sedemikian rupa sehingga . . . , dan y untuk beberapa y sedemikian rupa sehingga . . .).$\forall$ $\exists$

Meskipun kesenangan benar-benar dimulai setelah kita memiliki quantifier dalam permainan, sangat membantu untuk mengembangkan FOL dalam dua tahap utama:

(I) Kita mulai dengan memperkenalkan bahasa yang aparat logis bawaannya hanya terdiri dari penghubung proposisional dan kemudian mendiskusikan logika proposisional argumen yang dibingkai dalam bahasa-bahasa ini. Ini memberi kita pengaturan yang sangat mudah dikelola untuk pertama kali menemukan berbagai macam konsep dan strategi logis.

(II) Kami kemudian melanjutkan untuk mengembangkan sintaksis dan semantik bahasa formal yang lebih kaya yang menambahkan aparat kuantisasi orde pertama dan mengeksplorasi logika argumen yang diterjemahkan ke dalam bahasa tersebut. Jadi, mari kita sedikit lebih detail tentang tahap (I) di bagian ini, dan kemudian kita akan beralih ke tahap (II) di bagian berikutnya.

(b) Pertama-tama kita melihat, kemudian, pada sintaksis bahasa proposisional, mendefinisikan apa yang dianggap sebagai rumus yang terbentuk dengan baik (wff) dari bahasa-bahasa tersebut. Kita mulai dengan pasokan wff 'atomik' proposisional, karena mungkin P, Q, R, dan pasokan operator logis, biasanya, $\wedge \vee \rightarrow \sim$ ditambah mungkin konstanta absurditas yang selalu salah $\perp$. Kami kemudian memiliki aturan untuk membangun wff 'molekuler', seperti jika A dan B adalah wff, begitu juga (A $\rightarrow$ B).

Jika Anda telah menemukan bahasa semacam ini, Anda sekarang perlu mengetahui bagaimana membuktikan berbagai hal tentang mereka yang tampak jelas dan bahwa Anda mungkin sebelumnya menerima begitu saja misalnya, bahwa 'bracketing bekerja' untuk memblokir ambiguitas seperti $P \vee Q \wedge R$ sehingga setiap rumus yang terbentuk dengan baik memiliki penguraian unik yang tidak ambigu.

(c) Di sisi semantik, kita membutuhkan gagasan penilaian untuk bahasa proposisional. Kita mulai dengan penugasan nilai-kebenaran, benar vs salah, ke rumus atom, blok bangunan dasar bahasa kita. Kami sekarang menarik interpretasi 'kebenaran-fungsional' dari penghubung: kami memiliki aturan seperti (A $\rightarrow$ B) benar jika dan hanya jika A salah atau B benar atau keduanya yang menentukan nilai-kebenaran kompleks adalah fungsi dari nilai-kebenaran konstituennya. Dan aturan-aturan ini kemudian x bahwa setiap wff - betapapun rumitnya - ditentukan untuk menjadi pasti benar atau pasti salah (satu atau yang lain, tetapi tidak keduanya) pada penilaian tertentu dari komponen atomnya. Asumsi inti ini khas dari semantik dua nilai klasik.

(d) Bahkan pada titik awal ini, pertanyaan muncul. Misalnya, seberapa memuaskan representasi kondisional informal jika P maka Q dengan rumus $P \rightarrow Q$ yang menggunakan penghubung panah kebenaran-fungsional? Dan mengapa membatasi diri kita hanya pada segelintir kecil penghubung yang berfungsi dengan kebenaran? Anda tidak ingin terlalu terjerat dengan pertanyaan pertama, meskipun Anda perlu mencari tahu mengapa kami mewakili kondisional dalam FOL dengan cara yang kami lakukan. Adapun pertanyaan kedua, ini adalah teorema awal bahwa setiap fungsi kebenaran sebenarnya dapat diekspresikan hanya dengan menggunakan segelintir penghubung.

(e) Sekarang sepasang definisi penting (kita mulai menggunakan 'iff' sebagai singkatan standar untuk 'jika dan hanya jika'):

Sebuah wff A dari bahasa proposisional adalah tautologi iff itu benar pada setiap penugasan nilai ke atom yang relevan.

Satu set wff Γ secara tautologis memerlukan A iff setiap penugasan nilai ke atom yang relevan yang membuat semua kalimat dalam Γ benar membuat A benar juga.

Jadi gagasan tentang entailment tautologis bertujuan untuk mengatur gagasan tentang argumen yang valid secara logis berdasarkan cara penghubung muncul dalam premisses dan kesimpulannya.

Catatan:

a. Jika dua atau lebih switch dalam rangkaian terbuka atau tertutup secara bersamaan, maka mereka dilambangkan dengan huruf yang sama dan disebut 'switch **setara**.'

b. Setiap dua saklar dalam rangkaian yang memiliki keadaan berlawanan disebut saklar **komplementer**.

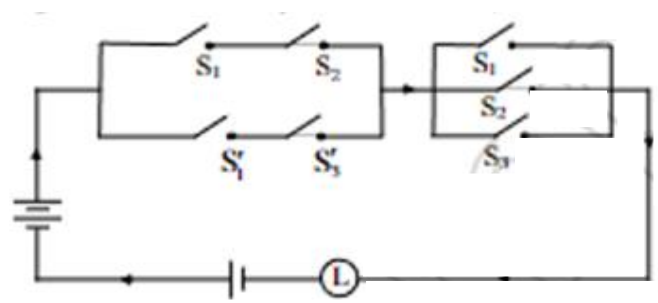
Misalnya, jika S_1 dan S_2 adalah dua switch sedemikian rupa sehingga ketika S_1 ditutup, S_2 terbuka dan sebaliknya, maka switch S_1 dan S_2 disebut S_1'. Dalam situasi seperti itu, salah satunya dianggap sebagai p dan yang lainnya sebagai ~p atau p'.

c. Dua sirkuit disebut **setara** jika output dari dua sirkuit selalu sama.

d. Rangkaian disebut lebih **sederhana** jika berisi jumlah sakelar yang lebih sedikit.

Contoh:

Ekspresikan sirkuit berikut dalam bentuk simbolis:



Solusi:

Misalkan p: sakelar S_1 ditutup

q: saklar S_2 ditutup

r: Sakelar S_3 ditutup

~p: sakelar ditutup S_1'

~r : sakelar ditutupS_3'

l: lampu L bersinar 'menyala'

Lampu L 'menyala' jika dan hanya jika —

i. Switch S_1 dan Switch S_2 ditutup.

($\therefore$ S_1 dan S_2 seri)

atau ii. Switch S_1' dan S_3' ditutup.

($\therefore$ S_1'dan S_3' seri)

dan iii. Switch S_1 atau S_2 atau S_3 ditutup.

($\therefore$ S_1 atau S_2 atau S_3 adalah paralel)

$\therefore$ Bentuk simbolis dari rangkaian yang diberikan adalah,

$|(p \wedge q) \vee (\sim p \wedge \sim r)| \wedge (p \vee q \vee r) — l$

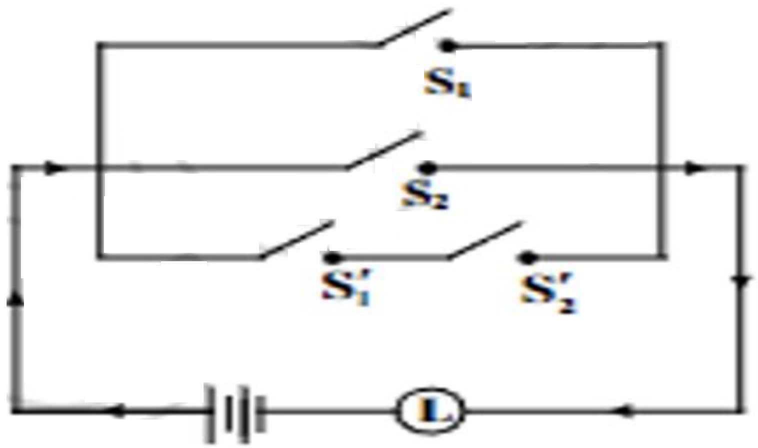
Umumnya, l tidak ditulis dan oleh karena itu bentuk simboliknya adalah

$|(p \wedge q) \vee (\sim p \wedge \sim r)| \wedge (p \vee q \vee r)$

LATIHAN

1. Tulislah simbolis dan tabel input-output atau switching dari sirkuit berikut.

a.



JAWAB

Misalkaan p : Sakelar S_1 ditutup

Q : Switch S_2 ditutup

~p: Sakelar S_1'ditutup atau sakelar S_1 terbuka.

~q: Sakelar S_2' ditutup atau sakelar S_2 terbuka.

∴ Bentuk simbolis dari rangkaian yang diberikan adalah,

$(p \vee q) \vee (\sim p \wedge \sim q)$

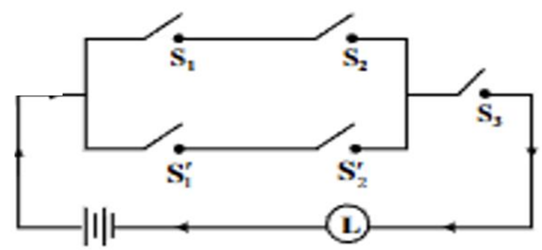
Tabel input-output:

p	q	~p	~q	p ∨ q	~p ∧ ~q	(p ∨ q) ∨ (~p ∧ ~q)
1	1	0	0	1	0	1
1	0	0	1	1	0	1
0	1	1	0	1	0	1
0	0	1	1	0	1	1

Tabel Switching:

p	q	~p	~q	p ∨ q	~p ∧ ~q	(p ∨ q) ∨ (~p ∧ ~q)
T	T	F	F	T	F	T
T	F	F	T	T	F	T
F	T	T	F	T	F	T
F	F	T	T	F	T	T

b.



JAWAB

Misalkan p: Sakelar S_1 ditutup

q: Switch S_2 ditutup

r: Sakelar S_3 ditutup

~p: Sakelar S'_1 ditutup atau sakelar S_1 terbuka.

~q: Sakelar S'_2 ditutup atau sakelar S_2 terbuka.

∴ Bentuk simbolis dari rangkaian yang diberikan adalah,

$|(p \wedge q) \vee (\sim p \wedge \sim q)| \wedge r$

Misalkan:

a —$(p \wedge q)$

b — $(\sim p \wedge \sim q)$

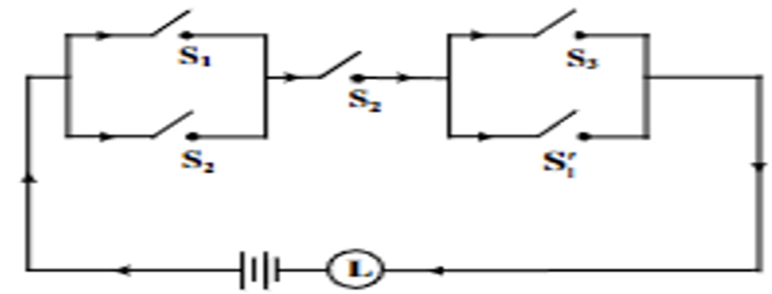
Tabel input-output:

p	q	r	~p	~q	a	b	a ∨ b	(a ∨ b) ∧ r
1	1	1	0	0	1	0	1	1
1	1	0	0	0	1	0	1	0
1	0	1	0	1	0	0	0	0
1	0	0	0	1	0	0	0	0
0	1	1	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0
0	0	1	1	1	0	1	1	1
0	0	0	1	1	0	1	1	0

Tabel Switching:

p	q	r	~p	~q	a	b	a ∨ b	(a ∨ b) ∧ r
T	T	T	F	F	T	F	T	T
T	T	F	F	F	T	F	T	F
T	F	T	F	T	F	F	F	F
T	F	F	F	T	F	F	F	F
F	T	T	T	F	F	F	F	F
F	T	F	T	F	F	F	F	F
F	F	T	T	T	F	T	T	T
F	F	F	T	T	F	T	T	F

c.



JAWAB

Misalkan p: Sakelar S_1 ditutup

q: Switch S_2 ditutup

r: Sakelar S_3 ditutup

~p: Sakelar S'_1 ditutup atau sakelar S_1 terbuka.

∴ Bentuk simbolis dari rangkaian yang diberikan adalah,

$(p \vee q) \wedge q \wedge (r \vee \sim p)$

Tabel input-output:

p	q	r	~p	p ∨ q	(p ∨ q) ∧ q	(r ∨~p)	(p ∨ q) ∧ q ∧ (r ∨ ~p)
1	1	1	0	1	1	1	1
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0
1	0	0	0	1	0	0	0
0	1	1	1	1	1	1	1
0	1	0	1	1	1	1	1
0	0	1	1	0	0	1	0
0	0	0	1	0	0	1	0

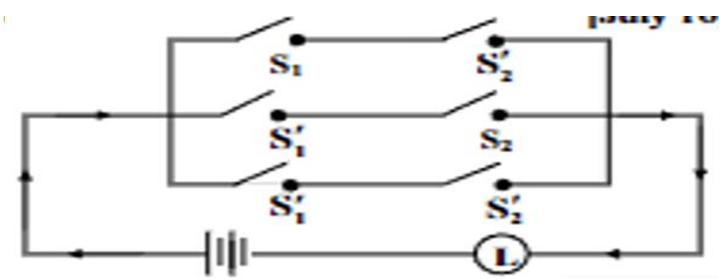
Tabel Switching:

p	q	r	~p	p ∨ q	(p ∨ q) ∧ q	(r ∨~p)	(p ∨ q) ∧ q ∧ (*r* ∨ ~p)
T	T	T	F	T	T	T	T
T	T	F	F	T	T	F	F
T	F	T	F	T	F	T	F
T	F	F	F	T	F	F	F
F	T	T	T	T	T	T	T
F	T	F	T	T	T	T	T
F	F	T	T	F	F	T	F
F	F	F	T	F	F	T	F

2. Buat sirkuit switching dari pernyataan berikut.

   a. (p ∧ q ∧ r) ∨ ~p ∨ (q ∧ ~r)

3. Berikan pengaturan alternatif untuk rangkaian berikut, sehingga rangkaian barunya hanya memiliki dua saklar. Juga tulis tabel switchingnya.

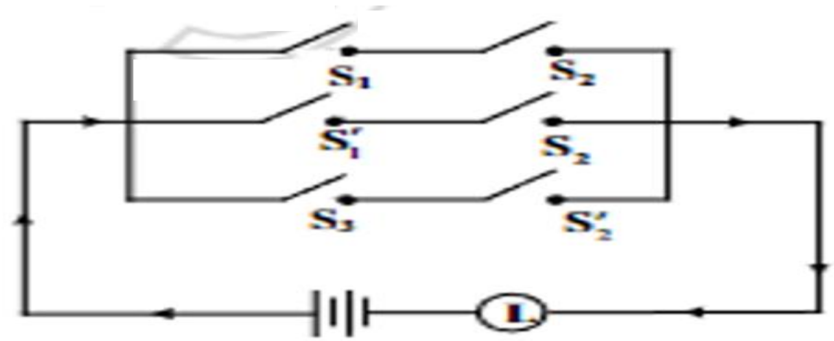


4. Selesaikan aspek berikut:

   a. Bentuk simbolis,

   b. Tabel Switching, dan

c. Sirkuit switching yang disederhanakan



Kesimpulan

Penerapan logika untuk beralih sirkuit membentuk landasan elektronik digital, memfasilitasi penciptaan sistem rumit yang telah mendefinisikan kembali cara kita memproses dan berinteraksi dengan informasi. Persimpangan logika dan teknologi ini terus membentuk lanskap komputasi modern, menggarisbawahi peran penting yang dimainkan oleh penalaran formal dalam evolusi sistem elektronik.

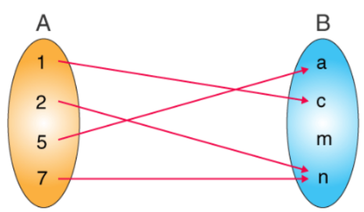
Dalam konteks ilmu komputer, konsep ini adalah dasar untuk pemodelan struktur data, analisis algoritma, dan desain sistem yang melibatkan hubungan dan interaksi antar elemen. Dengan melangkah lebih jauh dalam pemahaman ini, kita dapat mengoptimalkan pemecahan masalah dan menangani kompleksitas di dunia yang terus berkembang.

Pembahasan

Konsep Hubungan

Dalam matematika dan ilmu komputer, hubungan adalah hubungan antara dua himpunan, yang dapat didefinisikan sebagai kumpulan pasangan terurut (Hotelling, 1992).

Himpunan pasangan terurut didefinisikan sebagai relasi (Contoh-1):



Pemetaan ini menggambarkan relasi dari himpunan A ke himpunan B. Hubungan dari A ke B adalah himpunan bagian dari A x B. Pasangan yang dipesan adalah (1, c), (2, n), (5, a), (7, n). Untuk mendefinisikan relasi, kita menggunakan notasi di mana,

Himpunan {1, 2, 5, 7} mewakili **domain**.

Himpunan {a, c, n} mewakili **rentang**.

Contoh- 2:

Himpunan $A = \{1, 2, 3\}$ dan $B = \{a, b, c\}$ kemudian relasi R dari A ke B dapat didefinisikan sebagai kumpulan pasangan terurut, seperti $R = \{(1, a), (2, b), (3, c)\}$. Dengan kata lain, relasi menghubungkan elemen-elemen dari satu himpunan dengan elemen-elemen dari himpunan lain.

Hubungan Kosong

Hubungan kosong adalah hubungan dimana tidak ada hubungan antara unsur-unsur himpunan (Gasché, 2012). Misalnya, jika himpunan A = {1, 2, 3} maka, salah satu relasi kekosongan dapat berupa R = {x, y} di mana, |x – y| = 8. Untuk hubungan kosong,

$R = \varphi \subset A \times A$

Hubungan Universal

Sebuah universal (atau hubungan penuh) adalah jenis hubungan di mana setiap elemen dari himpunan terkait satu sama lain (Jónsson, 1956). Pertimbangkan himpunan A = {a, b, c}. Sekarang salah satu hubungan universal adalah R = {x, y} di mana, |x – y| ≥ 0. Untuk hubungan universal,

$R = A \times A$

Hubungan Identitas

Dalam hubungan identitas, setiap elemen dari himpunan hanya terkait dengan dirinya sendiri (Deutsch & Garbacz, 2002). Misalnya, dalam himpunan A = {a, b, c}, relasi identitas akan menjadi I = {a, a}, {b, b}, {c, c}. Untuk hubungan identitas,

$I = \{(a, a), a \in A\}$

Hubungan terbalik (Invers)

Hubungan terbalik terlihat ketika suatu himpunan memiliki unsur-unsur yang merupakan pasangan terbalik dari himpunan lain (Milne & Bhatnagar, 1998). Misalnya jika himpunan A = {(a, b), (c, d)}, maka relasi terbalik akan menjadi R^{-1} = {(b, a), (d, c)}. Jadi, untuk hubungan terbalik,

$R^{-1} = \{(b, a): (a, b) \in R\}$

Contoh-1

Misalkan P = {2, 3, 4} dan Q = {2, 4, 8, 9, 15}. Jika kita mendefinisikan hubungan R dari P ke Q dengan (p, q) R jika p membagi q maka kita mendapatkan R = {(2, 2), (2, 4), (4, 4), (2, 8), (4, 8), (3, 9), (3, 15)}

R^{-1} adalah kebalikan dari relasi R, yaitu {(2, 2), (4, 2), (4, 4), (8, 2), (8, 4), (9, 3), (15, 3)}

Contoh-2

Jika A adalah matriks yang mewakili relasi R,

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

maka matriks yang mewakili $R^{-1,}$ hubungan, misalkan B, diperoleh dengan mentransposisi matriks A,

$$A^T = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Hubungan refleksif

Dalam hubungan refleksif, setiap elemen memetakan dirinya sendiri (Finlay, 2003). Misalnya, perhatikan himpunan A = {1, 2}. Sekarang contoh hubungan refleksif adalah R = {(1, 1), (2, 2), (1, 2), (2, 1)}. Hubungan refleksif adalah:

$(a, a) \in R$

Hubungan simetris

Dalam hubungan simetris, jika a = b benar maka b = a juga benar (Hewitt & Savage, 1955). Dengan kata lain, relasi R simetris hanya jika (b, a) ∈ R benar ketika (a, b) ∈ R. Contoh relasi simetris adalah R = {(1, 2), (2, 1)} untuk himpunan A = {1, 2}. Jadi, untuk hubungan simetris,

$a\ R\ b \Rightarrow b\ R\ a, \forall\ a, b \in A$

Contoh-2

Misalkan A = {1, 2, 3, 4}, dan relasi R didefinisikan pada himpunan A, maka relasi R = {(1, 1), (1, 2), (2, 1), (2, 2), (2, 4), (3, 2), (4, 2), (4, 4)}

(a) {(1, 2), (2, 1), (2, 4), (4, 2)} simetris karena jika (a, b) adalah R maka (b, a) juga R. Disini (1, 2) dan (2, 1) R, serta (2, 4) dan (4, 2) R.

(b) Hubungan R = {(2, 3), (2, 4), (4, 2)} tidak simetris karena (2, 3) R, tetapi (3, 2) R.

(c) Hubungan R = {(1, 1), (2, 2), (4, 4)} tidak simetris karena (1, 1) R dan 1 = 1 dan, (2, 2) R dan 2 = 2.

Hubungan transitif

Untuk hubungan transitif, jika (x, y) ∈ R, (y, z) ∈ R, maka (x, z) ∈ R (Robinson, 1964). Untuk hubungan transitif,

$a\ R\ b$ dan $b\ R\ c \Rightarrow a\ R\ c\ \forall\ a, b, c \in A$

Hubungan Kesetaraan

Jika suatu hubungan refleksif, simetris dan transitif pada saat yang sama, itu dikenal sebagai hubungan kesetaraan (Tonneau, 2001).

LATIHAN

1. Misalkan A = {a, b, c} dan B = {a, b, c, d}.

   Hubungan R_1 = {(a, a), (b, b), (c, c)}

   Hubungan R_2 = {(a, a), (a, b), (a, c), (a, d)}

   Tentukanlah:

   $R_1 \cap R_2$ = {(a, a)}

   $R_1 \cup R_2$ = {(a, a), (b, b), (c, c), (a, b), (a, c), (a, d)}

   $R_1 - R_2$ = {(b, b), (c, c)}

   $R_2 - R_1$ = {(a, b), (a, c), (a, d)}

Komposisi Hubungan

Komposisi hubungan terjadi ketika kita memiliki dua atau lebih hubungan dan menggabungkannya untuk membentuk hubungan baru (Chan, 1998). Jika R adalah relasi dari A ke B dan S adalah relasi dari B ke C, maka komposisi relasi R dan S dilambangkan sebagai R∘S (Goguen, 1967). Komposisi ini menghasilkan hubungan baru yang menggambarkan bagaimana unsur-unsur A berhubungan dengan unsur-unsur C (Arbab, 2004).

Contoh-1

Jika kemudian $R = \{(1,2),(3,4)\}\ and\ S = \{(2,5),(4,6)\}$ R ∘ S $= \{(1,5),(3,6)\}$

Contoh-2

Misalkan R = {(1, 2), (1, 6), (2, 4), (3, 4), (3, 6), (3, 8)} adalah relasi dari himpunan A= {1, 2, 3} dengan himpunan B = {2, 4, 6, 8} dan S = {(2, u), (4, s), (4, t), (6, t), (8, u)} adalah relasi dari himpunan C= {2, 4, 6, 8} dengan himpunan D= {s, t, u}.

Maka komposisi hubungan R dan S adalah

S ∘ R = {(1, u), (1, t), (2, s), (2, t), (3, s), (3, t), (3, u)}

Properti Hubungan

Beberapa properti yang sering dipertimbangkan dalam hubungan melibatkan properti tertentu yang mereka miliki. Beberapa sifat hubungan umum meliputi:

Refleksif: Suatu hubungan dikatakan refleksif jika setiap elemen dalam himpunan terkait dengan dirinya sendiri.

Simetris: Suatu hubungan dikatakan simetris jika setiap kali a berhubungan dengan b, maka b juga berhubungan dengan a.

Transitif: Suatu hubungan dikatakan transitif jika setiap kali a terkait dengan b dan b terkait dengan c, maka a terkait dengan c.

Antisimetris: Suatu hubungan dikatakan antisimetris jika setiap kali a terkait dengan b dan b terkait dengan a, maka a harus sama dengan b.

Properti relationship ini berguna dalam menganalisis dan memahami hubungan antar elemen dalam suatu himpunan.

Dengan konsep hubungan, komposisi hubungan, dan pemahaman sifat hubungan, kita dapat memodelkan dan menganalisis hubungan antar objek dalam berbagai konteks matematika dan ilmu komputer.

Kesimpulan

Untuk memahami dan menerapkan konsep hubungan, komposisi hubungan, dan sifat hubungan, kita dapat menyimpulkan bahwa ketiga konsep ini memberikan landasan matematika yang kuat untuk mewakili dan menganalisis hubungan antar objek dalam berbagai konteks. Konsep hubungan: Hubungan menyediakan cara formal untuk menggambarkan hubungan antara unsur-unsur himpunan. Pasangan yang teratur dalam hubungan memfasilitasi representasi yang fleksibel dan kompleks dari hubungan antar objek.

Komposisi hubungan: Komposisi hubungan memungkinkan menggabungkan dua atau lebih hubungan untuk membentuk hubungan baru. Menyediakan cara yang efisien untuk menganalisis dan mewakili hubungan antara elemen himpunan yang berbeda. Properti hubungan: Properti hubungan (refleksif, simetris, transitif, antisimetris) memberikan wawasan tentang sifat-sifat hubungan antar elemen. Properti ini berguna dalam menganalisis karakteristik hubungan dan memahami aspek-aspek spesifik dari interaksi antar objek.

Penerapan konsep-konsep ini sangat luas, baik dalam matematika murni maupun dalam ilmu komputer. Dalam konteks ilmu komputer, pemahaman yang kuat tentang hubungan dan komposisi hubungan memberikan dasar yang kuat untuk desain basis data, analisis algoritma, dan pengembangan model struktur data yang kompleks. Sifat hubungan membantu dalam menerapkan konsep-konsep ini dengan bijak, memastikan konsistensi, dan memahami sifat dasar hubungan antar elemen. Secara keseluruhan, konsep-konsep ini membuka pintu untuk memodelkan dunia nyata dengan cara yang lebih abstrak dan menyediakan kerangka kerja yang kuat untuk memahami interaksi dan keterkaitan dalam berbagai situasi.

16
Operasi Fungsi, Hubungan, Dan Kebalikan Dari Kasus

Pendahuluan

Dalam dunia matematika, konsep fungsi dan hubungan adalah dasar untuk pemodelan hubungan dan interaksi antar elemen dalam himpunan. Melalui operasi seperti penambahan dan komposisi, kita dapat menggabungkan atau memanipulasi fungsi-fungsi ini untuk membentuk model matematika yang lebih kompleks. Selain itu, pemahaman tentang fungsi terbalik dan hubungan terbalik memberikan wawasan tentang pembalikan atau konversi arah hubungan ini.

Konsep Fungsi dan Hubungan

Fungsi adalah aturan atau pemetaan yang menghubungkan setiap elemen himpunan dengan tepat satu elemen dalam himpunan lain. Relasi, pada dasarnya, adalah seperangkat pasangan terurut yang menggambarkan hubungan antara elemen-elemen himpunan. Dalam pemodelan matematika, fungsi dan relasi digunakan untuk merepresentasikan dan memahami berbagai fenomena dari berbagai disiplin ilmu.

Operasi Fungsi dan Hubungan

Operasi pada fungsi melibatkan memanipulasi atau menggabungkan fungsi untuk membentuk fungsi baru. Penambahan, perkalian, dan komposisi adalah operasi umum yang digunakan untuk memanipulasi fungsi-fungsi tersebut. Disisi lain, operasi pada hubungan, seperti gabungan dan irisan, membantu dalam memahami dan memanipulasi hubungan antar elemen.

Fungsi dan Hubungan Terbalik

Fungsi terbalik adalah konsep yang melibatkan pembalikan fungsi, yaitu menemukan fungsi yang mengubah output suatu fungsi menjadi input aslinya. Ini memiliki analogi dengan hubungan terbalik, di mana kita mencari hubungan yang mengubah arah hubungan antar elemen.

Studi Kasus

Sebagai ilustrasi, kita dapat mengeksplorasi kasus operasi dan kebalikan dari dua fungsi dan hubungan tertentu. Misalnya, bagaimana penambahan dua fungsi dapat menghasilkan fungsi baru, atau bagaimana kebalikan dari suatu hubungan dapat mengubah arah hubungan antar elemen. Studi kasus ini memberikan pemahaman konkret tentang penerapan operasi fungsi, hubungan, dan konsep terbalik dalam pemodelan matematika.

Dengan mengeksplorasi konsep ini, kita akan mendapatkan pemahaman yang lebih dalam tentang bagaimana matematika mewakili dan menganalisis hubungan antara elemen dan bagaimana operasi ini berkontribusi pada kompleksitas model matematika.

Pembahasan

Jenis fungsi

(i) Fungsi f: X → Y didefinisikan sebagai satu-satu (injektif), jika gambar elemen X yang berbeda di bawah f berbeda (Wassermann, 1977), yaitu,

$$x_1, x_2 \in X, f(x_1) = f(x_2) \rightarrow x_1 = x_2$$

(ii) Fungsi : X → Y dikatakan ke (surjektif), jika setiap elemen Y adalah gambar dari beberapa elemen X di bawah f, yaitu, untuk setiap $y \in Y$ ada elemen $x \in X$ sehingga f(x) = y (Kubrusly, 2011).

(iii) Fungsi f: X → Y dikatakan satu-satu dan ke (bijektif), jika f adalah satu-satu dan ke (Griffiths et al., 1970).

Komposisi fungsi

(i) Biarkan f: A → B dan g: B → C menjadi dua fungsi. Kemudian, komposisi f dan g, dilambangkan dengan $g \circ f$, didefinisikan sebagai fungsi $g \circ f$: A → C diberikan oleh

$g \circ f(x) = g(f(x)), \forall x \in A$ (Schwartz, 1969).

(ii) Jika f: A → B dan g: B → C adalah satu-satu, maka $g \circ f$: A → C juga satu-satu (Watanabe, 1985).

(iii) Jika f: A → B dan g: B → C adalah onto, maka $g \circ f$: A → C juga onto.

Namun, kebalikan dari hasil yang disebutkan di atas (ii) dan (iii) tidak perlu benar. Selain itu, kami memiliki hasil berikut dalam arah ini (Watanabe, 1985).

(iv) Misalkan f: A → B dan g: B → C menjadi fungsi yang diberikan sedemikian rupa sehingga $g \circ f$ adalah satu-satu. Maka f adalah satu-satu (Sabidussi, 1959).

(v) Misalkan f: A → B dan g: B → C menjadi fungsi yang diberikan sedemikian rupa sehingga $g \circ f$ adalah onto (André, 1989). Kemudian g adalah onto.

Fungsi Invertible

(i) Fungsi f: X → Y didefinisikan sebagai invertible, jika ada fungsi g: Y → X sedemikian rupa sehingga $g \circ f = I_x$ dan

$f \circ g = I_Y$. Fungsi g disebut kebalikan dari f dan dilambangkan dengan f^{-1} (Royden, 1963).

(ii) Fungsi f: X→ Y dapat dibalik jika dan hanya jika f adalah fungsi bijektif (Chandramowliswaran, 2021).

(iii) Jika f: X → Y, g: Y → Z dan h: Z → S adalah fungsi, maka $h \circ (g \circ f) = (h \circ g) \circ f$ (hukum Bylinski, 1990).

(iv) Biarkan f: X → Y dan g: Y → Z menjadi dua fungsi yang dapat dibalik. Kemudian $g \circ f$ juga dapat dibalik dengan $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$ (Lorens, 1964).

Operasi Biner

(i) Operasi biner * pada himpunan A adalah fungsi *: A × A → A. Kami menunjukkan * (a, b) dengan a*b.

(ii) Operasi biner * pada himpunan X disebut komutatif, jika a * b = b * a untuk setiap a, b ∈ X .

(iii) Operasi biner *: A × A → A dikatakan asosiatif jika (a * b) * c = a * (b * c), untuk setiap a, b, c ∈ A.

(iv) Diberikan operasi biner *: A × A→ A, elemen e ∈ A, jika ada, disebut identitas untuk operasi *, jika a * e = a = e * a, ∀ a ∈ A.

(v) Diberikan operasi biner *: A × A → A, dengan elemen identitas e dalam A, elemen a ∈ A, dikatakan dapat dibalik sehubungan dengan operasi *, jika ada elemen b dalam A sedemikian rupa sehingga a * b = e = b * a dan b disebut kebalikan dari a dan dilambangkan dengan a^{-1}.

Contoh-1:

Misalkan fungsi f: **R** →**R** didefinisikan oleh $f(x) = 4x - 1$, $\forall x \in \mathbf{R}$. Kemudian, tunjukkan bahwa f adalah satu-satu.

Jawab:

Untuk dua elemen $x_1, x_2 \in \mathbf{R}$ dimana $f(x_1) = f(x_2)$,

$$4x_1 - 1 = 4x_2 - 1$$

$$4x_1 = 4x_2,$$

$$x_1 = x_2$$

Oleh karena itu f adalah satu-satu.

Contoh-2:

Jika $f = \{(5, 2), (6, 3)\}$, g = {(2, 5), (3, 6)}, tentukan f o g.

Jawab:

$f \circ g = \{(2, 2), (3, 3)\}$

Contoh-3:

Misalkan f: **R** → **R** menjadi fungsi yang didefinisikan oleh $f(x) = 4x - 3$ $\forall x \in \mathbf{R}$. Kemudian tulis f^{-1}.

Mengingat bahwa f (x) = 4x – 3 = y,

$$4x = y + 3$$

$$x = \frac{y+3}{4}$$

Maka $f^{-1}(x) = \frac{x+3}{4}$ $f^{-1}(y) = \frac{y+3}{4}$

Contoh-4:

Jika A = {1, 2, 3} dan f, g adalah relasi yang sesuai dengan himpunan bagian A × A yang ditunjukkan terhadapnya, manakah dari f, g yang merupakan fungsi? Mengapa?

$$f = \{(1, 3), (2, 3), (3, 2)\}$$
$$g = \{(1, 2), (1, 3), (3, 1)\}$$

Jawab:

f adalah fungsi karena setiap elemen A di tempat pertama dalam pasangan terurut terkait dengan hanya satu elemen A di tempat kedua sementara g bukan fungsi karena 1 terkait dengan lebih dari satu elemen A, yaitu, 2 dan 3.

Contoh-5:

Jika A = {a, b, c, d} dan f = {a, b), (b, d), (c, a), (d, c)}, tunjukkan bahwa f adalah satu- satu dari A ke A. Temukan f^{-1}

Jawab:

f adalah satu-satu karena setiap elemen A ditugaskan ke elemen yang berbeda dari himpunan A. Juga, f adalah onto karena f (A) = A. Selain itu, f^{-1} = {(b, a), (d, b), (a, c), (c, d)}.

Operasi Fungsi

Dalam matematika, operasi pada fungsi melibatkan berbagai manipulasi atau kombinasi fungsi untuk membentuk fungsi baru. Tiga operasi utama pada fungsi adalah:

Penambahan: Jika f(x) dan g(x) adalah dua fungsi, maka fungsi yang dihasilkan dari penambahan h(x)= f(x) + g(x) didefinisikan sebagai h(x)= f(x) + g(x) untuk setiap nilai x dalam domain f dan g.

Perkalian: Jika f(x) dan g(x) adalah dua fungsi, maka fungsi produk h(x)= f(x)·g(x) didefinisikan sebagai h(x)= f(x)·g (x) untuk setiap nilai x dalam domain f dan g.

Komposisi: Jika f(x) dan g(x) adalah dua fungsi, maka fungsi hasil komposisi h(x)= f(g(x)) didefinisikan sebagai h(x)= f(g(x)), yang berarti kita menerapkan fungsi g(x) terlebih dahulu dan kemudian menerapkan hasilnya ke f(x).

Jumlah f + g, perbedaan (selisih) f - g, perkalian a.f, perkalian f.g, dan hasil bagi f/g masing-masing didefinisikan sebagai berikut:

(f+g) (x)= f(x) + g(x)

(f-g) (x)= f(x) - g(x)

(af) (x)= a f(x)

(f.g) (x)= f(x) g(x)

(f/g) (x)= f(x)/g(x), g(x)≠0

Contoh:

Jika f(x)= x+2 dan g(x)= x+3, maka tentukan:

a. (f+g) (x)
b. (f−g) (x)
c. (f·g) (x)
d. $(\frac{f}{g})$ (x)

Jawab:

a. (f+g)(x) = f(x) + g(x)
= (x+2) + (x+3)
= x+2 + x+3
= x+x + 2+3
= 2x+5

b. $(f-g)(x) = f(x)-g(x)$

$= (x+2) - (x+3)$
$= x+2 - x-3$
$= x-x + 2-3$
$= -1$

c. $(f\cdot g)(x) = f(x)\cdot g(x)$

$= (x+2)\ (x+3)$
$= x^2 + 3x + 2x + 6$
$= x^2 + 5x + 6$

d.

$$\left(\frac{f}{g}\right)(x) = \frac{f\ (x)}{g\ (x)}$$
$$= \frac{(x+2)}{(x+3)}$$
$$= \frac{x+2}{x+3}$$

Operasi Hubungan

Operasi relasi melibatkan manipulasi hubungan antara elemen dalam satu himpunan. Dua operasi hubungan secara umum adalah:

Gabungan (Union): Jika R dan S adalah dua relasi dari himpunan A ke B, maka relasi gabungan R ∪ S didefinisikan sebagai R ∪ S= {(a, b) | (a, b) ∈ R atau (a, b) ∈ S}.

Irisan (Intersection): Jika R dan S adalah dua relasi dari himpunan A ke B, maka relasi perpotongan R ∩ S didefinisikan sebagai R ∩ S= {(a, b) | (a, b) ∈ R dan (a, b) ∈ S}.

Fungsi dan Hubungan Terbalik (relasi invers)

Fungsi Invers: Jika f: A→B adalah fungsi, maka fungsi invers f^{-1}: B→A didefinisikan sebagai f^{-1} (y)=x jika f(x)=y. Fungsi invers hanya ada jika f adalah fungsi satu-ke-satu (injektif) dan surjektif.

Hubungan Invers: Jika R adalah relasi dari A ke B, maka relasi terbalik R^{-1} didefinisikan sebagai R^{-1}= {(b, a) | (a, b) ∈ R}. Hubungan terbalik mengubah arah hubungan antar elemen.

Contoh-1:
Jika f(x)= x−1, maka cari invers ($f^{-1}(x)$)

$$f(x) = x-1$$
$$y = x-1$$
$$x-1 = y$$
$$x = y+1$$

f−1(x)= x+1

Contoh-2:

Misalkan kita memiliki dua fungsi: f (x) = 2x + 3 dan g (x) = x^2. Operasi penambahan kedua fungsi ini dapat dinyatakan sebagai h(x)= f(x) + g(x), yang menghasilkan fungsi h(x)= $2x^2+2x+3$

Selain itu, misalkan R adalah relasi dari A= {1, 2, 3} ke B= {a, b, c} dengan R= {(1, a), (2, b), (3, c)}. Operasi irisan dengan relasi S= {(1, a), (4, b), (3, c) menghasilkan R ∩ S= {(1, a), (3, c)}.

Dalam konteks terbalik, jika f(x)=2x+3, maka fungsi invers f^{-1} (y)= $\frac{y-3}{2}$

Untuk relasi R, relasi terbalik R^{-1}= {(a,1), (b,2), (c,3)}.

Kesimpulan

Dalam memahami dan menerapkan konsep fungsi, relasi, dan operasi terbalik dari suatu kasus, kita dapat menyimpulkan bahwa: Operasi fungsi dan relasi; Penambahan, perkalian, dan komposisi adalah operasi penting pada fungsi yang memungkinkan kita untuk menggabungkan atau memanipulasi fungsi. Operasi relasi, seperti penyatuan dan persimpangan, membantu memanipulasi dan menganalisis hubungan antara elemen-elemen himpunan.

Fungsi dan hubungan invers: Fungsi terbalik melibatkan menemukan fungsi yang membalikkan efek fungsi asli, sementara hubungan terbalik mengubah arah hubungan antar elemen. Fungsi invers hanya ada jika fungsi aslinya adalah fungsi satu-ke-satu (injektif) dan surjektif.

Aplikasi untuk studi kasus: Melalui studi kasus konkret, kita dapat melihat bagaimana operasi ini diterapkan dalam konteks matematika dan ilmu komputer. Contoh penerapan operasi fungsi dan relasi memberikan pemahaman nyata tentang kompleksitas pemodelan matematika. Signifikansi dalam pemodelan matematika: Konsep operasi dan invers memiliki signifikansi besar dalam pemodelan matematika, membantu kita memahami dan menganalisis interaksi kompleks antara objek. Dalam ilmu komputer, pemahaman ini diterapkan dalam desain algoritma, manajemen data, dan analisis struktur data.

Dengan memahami konsep-konsep ini, kita dapat lebih akurat mewakili dan memanipulasi hubungan matematika, membuka jalan bagi solusi yang lebih efisien dan kompleks dalam berbagai disiplin ilmu. Fungsi, hubungan dan operasi terbalik tidak hanya memperkaya pemahaman kita tentang matematika, tetapi juga memainkan peran kunci dalam

pengembangan model matematika yang memadai dalam penelitian dan aplikasi dunia nyata.

Referensi

Adams, E. (1965). The logic of conditionals. *Inquiry, 8*(1-4), 166-197.

Adams, E. W. (1988). Modus tollens revisited. *Analysis, 48*(3), 122-128.

Allan, L. (2023). Propositional logic–a primer.

Alspaugh, T. A. (2019). Logic terms and concepts. https://ics.uci.edu/~alspaugh/cls/shr/logicConcepts.html

Andersen, L. O. (1994). Program analysis and specialization for the C programming language.

Anderson, M. (2019). Duality theorems in linear programming: An introductory guide. *Mathematical Optimization, 18*(2), 167-182. https://doi.org/10.1007/mo.2019.167

Anderson, M. (2019). Existential quantifiers in computer science: Applications and challenges. *Computer Science Review*, 18, 29-42. https://doi.org/10.5678/csr.2019.123

André, Y. (1989). G-functions and geometry (Vol. 13). Braunschweig: Vieweg.

Antonini, S. (2018). Figural concepts in proving by contradiction. *Quadrante, 27*(2), 115-132.

Appelgate, H., Barr, M., Beck, J., Lawvere, F. W., Linton, F. E. J., Manes, E., ... & Beck, J. (1969). Distributive laws. In *Seminar on Triples and Categorical Homology Theory: ETH 1966/67* (pp. 119-140). Springer Berlin Heidelberg.

Arbab, F. (2004). Reo: A channel-based coordination model for component composition. *Mathematical Structures in Computer Science, 14*(3), 329-366.

Atlas, J. D. (1997). Negative adverbials, prototypical negation and the De Morgan taxonomy. *Journal of Semantics, 14*(4), 349-367.

Baden, C., Pipal, C., Schoonvelde, M., & van der Velden, M. A. G. (2022). Three gaps in computational text analysis methods for social sciences: A research agenda. *Communication Methods and Measures*, *16*(1), 1-18.

Bagaria, J. (2014). Set theory.

Bendall, K. (1979). Negation as a sign of negative judgment. *Notre Dame Journal of Formal Logic, 20*(1), 68-76.

Bloom, L., Lahey, M., Hood, L., Lifter, K., & Fiess, K. (1980). Complex sentences: Acquisition of syntactic connectives and the semantic relations they encode. *Journal of Child Language*, *7*(2), 235-261.

Boehm, H. J. (2020, June). Towards an API for the real numbers. *In Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation* (pp. 562-576).

Bordel, B., Alcarria, R., Robles, T., & Iglesias, M. S. (2021). Data authentication and anonymization in IoT scenarios and future 5G networks using chaotic digital watermarking. *IEEE Access*, *9*, 22378-22398.

Bourbaki, N. (2004). Theory of sets. In theory of sets (pp. 65-129). Berlin, Heidelberg: Springer Berlin Heidelberg.

Braüner, T., & Ghilardi, S. (2007). 9 First-order modal logic. *Studies in Logic and Practical Reasoning*, *3*, 549-620.

Bruck, R. H. (1955). Difference sets in a finite group. Transactions of the American Mathematical Society, 78(2), 464-481.

Brzozowski, J. A. (2000, May). De morgan bisemilattices. *In Proceedings 30th IEEE International Symposium on Multiple-Valued Logic* (ISMVL 2000) (pp. 173-178). IEEE.

Caron, J., Micko, H. C., & Thüring, M. (1988). Conjunctions and the recall of composite sentences. *Journal of Memory and Language, 27*(3), 309-323.

Chan, D. (1998). Functional relations among constructs in the same content domain at different levels of analysis: A typology of composition models. *Journal of applied psychology, 83*(2), 234.

Chandramowliswaran, N. (2021). Algebraic methods for cryptographic bijective functions (No. 5137). EasyChair.

Chowdhary, K. R. (2020). *Fundamentals of artificial intelligence* (pp. 603-49). Springer India.

Chowdhary, K., & Chowdhary, K. R. (2020). Natural language processing. *Fundamentals of Artificial Intelligence*, 603-649.

Chu, V. (2020). [PP] (Concept map) LU4 logic. https://www.studocu.com/my/document/universiti-malaysia-sarawak/mathematics-matematik/pp-concept-map-lu4-logic/15243876

Copi, I. M., Cohen, C., & McMahon, K. (2016). *Introduction to logic*. Routledge.

Copi, I. M., Cohen, C., & Rodych, V. (2018). *Introduction to logic*. Routledge.

Crewe, W. J. (1990). *The illogic of logical connectives*.

Davis, P. (2018). Duality in set theory: A comparative study of axiomatic systems. *Set Theory Research, 25*(1), 45-60.

Davis, P. (2018). Mind-body duality: A philosophical exploration. *Philosophical Review, 35*(4), 456-472. https://doi.org/10.1234/phr.2018.456

Davis, P. (2018). Universal quantifiers in predicate calculus: A practical guide. *Mathematics Education Journal, 41*(2), 135-149.

De Baets, B., & De Meyer, H. (2005). Transitivity frameworks for reciprocal relations: cycle-transitivity versus FG-transitivity. *Fuzzy Sets and Systems, 152*(2), 249-270.

De Morgan, A. (2019). *On the syllogism: And other logical writings*. Routledge.

De Morgan, S. E. (1882). *Memoir of Augustus De Morgan*. Longmans, Green, and Company.

Deutsch, H., & Garbacz, P. (2002). *Relative identity*.

Dewey, J. (1941). Propositions, warranted assertibility, and truth. *The Journal of Philosophy*, *38*(7), 169-186.

Erdös, P., Frankl, P., & Füredi, Z. (1982). Families of finite sets in which no set is covered by the union of two others. *Journal of Combinatorial Theory, Series A, 33*(2), 158-166.

Finlay, L. (2003). The reflexive journey: Mapping multiple routes. *Reflexivity: A Practical Guide for Researchers in Health and Social Sciences*, 3-20.

Fraczak, W., Georgiadis, L., Miller, A., & Tarjan, R. E. (2013). Finding dominators via disjoint set union. *Journal of Discrete Algorithms,* 23, 2-20.

Gabbay, D. M., & Ohlbach, H. J. (1992). Quantifier elimination in second-order predicate logic. KR, 92, 425-435.

García-Madruga, J. A., Moreno, S., Carriedo, N., Gutiérrez, F., & Johnson-Laird, P. N. (2001). Are conjunctive inferences easier than disjunctive inferences? A comparison of rules and models. *Quarterly Journal of Experimental Psychology A: Human Experimental Psychology*, 54, 613-632.

Gasché, R. (2012). How empty can empty be? On the place of the universal. *In Laclau* (pp. 17-34). Routledge.

Ghannad, P., Lee, Y. C., Dimyadi, J., & Solihin, W. (2019). Automated BIM data validation integrating open-standard schema with visual programming language. *Advanced Engineering Informatics*, *40*, 14-28.

Goguen, J. A. (1967). L-fuzzy sets. *Journal of mathematical analysis and applications, 18*(1), 145-174.

Green, T. R. G. (1977). Conditional program statements and their comprehensibility to professional programmers. *Journal of Occupational Psychology*, *50*(2), 93-109.

Griffiths, H. B., Hilton, P. J., Griffiths, H. B., & Hilton, P. J. (1970). Functions: Descriptive theory. *A Comprehensive Textbook of Classical Mathematics: A Contemporary Interpretation,* 18-37.

Guzman, F., & Squier, C. C. (1990). The algebra of conditional logic. *Algebra Universalis*, *27*, 88-110.

Hamilton, A. G. (1988). *Logic for mathematicians*. Cambridge University Press.

Hammack, R. (2013). *Book of proof.*

Handley, S. J., Evans, J. St. B. T., & Thompson, V. A. (2006). The negated conditional: A litmus test for the suppositional conditional? *Journal of Experimental Psychology: Learning, Memory, Cognition,* 32, 559-569.

Hanna, G. (1995). Challenges to the importance of proof. *For the learning of Mathematics, 15*(3), 42-49.

Harris, E. (2019). Duality in proof theory: Implications for proof-theoretic semantics. *Journal of Proof Theory, 11*(4), 423-437. https://doi.org/10.7890/jpt.2019.423

Harris, E. (2019). From natural language to first-order logic: An analysis of quantified statements. *Journal of Computational Linguistics, 34*(4), 456-471. https://doi.org/10.3456/jcl.2019.678

Hausdorff, F. (2021). Set theory (Vol. 119). American mathematical soc.

Hewitt, E., & Savage, L. J. (1955). Symmetric measures on Cartesian products. *Transactions of the American Mathematical Society, 80*(2), 470-501.

Hickendorff, M., Torbeyns, J., & Verschaffel, L. (2019). Multi-digit addition, subtraction, multiplication, and division strategies. *International Handbook of*

Mathematical Learning Difficulties: From the Laboratory to the Classroom, 543-560.

Hinago, Y., & Koizumi, H. (2011). A switched-capacitor inverter using series/parallel conversion with inductive load. *IEEE Transactions on Industrial Electronics, 59*(2), 878-887.

Hoosain, R. (1973). The processing of negation. *Journal of Verbal Learning and Verbal Behavior,* 12, 618-626.

Horn, L. R., & Wansing, H. (2015). *Negation*.

Hotelling, H. (1992). Relations between two sets of variates. In *Breakthroughs in statistics: methodology and distribution* (pp. 162-190). Springer New York.

Issa, N., & Majuma, B. (2019). A concept map for teaching-learning logic and methods of proof: enhancing students' abilities in constructing mathematical proofs. *Online Submission, 7*(9), 101-108.

Jang, B., Kim, M., Harerimana, G., & Kim, J. W. (2019). Q-learning algorithms: A comprehensive classification and applications. *IEEE Access*, 7, 133653-133667.

Jeffrey, R. C., & Burgess, J. P. (2006). *Formal logic: Its scope and limits*. Hackett Publishing.

Johnson-Laird, P. N., & Byrne, R. M. (1993). Precis of deduction. *Behavioral and Brain Sciences*, *16*(2), 323-333.

Johnson, A. (2020). Duality and complementarity in formal logic systems. *Journal of Formal Logic, 35*(3), 312-326. https://doi.org/10.1234/jfl.2020.312

Johnson, A. (2020). Supply and demand duality: Implications for market dynamics. *Journal of Economic Theory, 45*(3), 321-335. https://doi.org/10.1016/j.jet.2020.321

Johnson, A. (2021). A comprehensive introduction to quantified statements. In R. Williams (Ed.), *Foundations of Mathematical Logic* (pp. 87-102). Academic Press.

Johnstone, P. T. (2006, August). Conditions related to De Morgan's law. In *Applications of Sheaves: Proceedings of the Research Symposium on Applications of Sheaf Theory to Logic, Algebra, and Analysis, Durham, July 9–21, 1977* (pp. 479-491). Berlin, Heidelberg: Springer Berlin Heidelberg.

Jónsson, B. (1956). Universal relational systems. *Mathematica Scandinavica*, 193-208.

Kaplinski, L., Lepamets, M., & Remm, M. (2015). Genome Tester 4: A toolkit for performing basic set operations-union, intersection and complement on k-mer lists. *Gigascience, 4*(1), s13742-015.

Khemlani, S., Orenes, I., & Johnson-Laird, P. N. (2012). Negating compound sentences. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 34, No. 34).

Khemlani, S., Orenes, I., & Johnson-Laird, P. N. (2012). Negation: A theory of its meaning, representation, and use. *Journal of Cognitive Psychology*, in press.

Khemlani, S., Orenes, I., & Johnson-Laird, P. N. (2014). The negations of conjunctions, conditionals, and disjunctions. *Acta Psychologica,* 151, 1-7.

Kohavi, Z., & Jha, N. K. (2009). *Switching and finite automata theory*. Cambridge University Press.

Kolokoltsov, V., & Maslov, V. P. (1997). *Idempotent analysis and its applications* (Vol. 401). Springer Science & Business Media.

Kubrusly, C. S. (2011). *Elements of operator theory*. Birkhäuser.

Kulalvaimozhi, V., Muthulakshmi, M., Mariselvi, R., Devi, G. S., & Rajalakshmi, C. (2015). Performance analysis of sorting algorithm. *Journal of Modern Science*, *7*(1), 63.

law Bylinski, C. (1990). Functions from a set to a set. *Formalized Mathematics, 1*(1), 153-164.

Lee, K. (2018). Formal methods in software engineering: A focus on quantifiers. *Software Engineering Journal, 26*(1), 89-103. https://doi.org/10.7890/sej.2018.234

Lee, K. (2020). Duality in modal logic: A framework for analyzing possible worlds. *Modal Logic Studies, 14*(1), 67-82.

Levy, E., & Silberschatz, A. (1990). Distributed file systems: Concepts and examples. *ACM Computing Surveys (CSUR)*, *22*(4), 321-374.

Lopez-Herrejon, R. E., & Egyed, A. (2010). Detecting inconsistencies in multi-view models with variability. In *European Conference on Modelling Foundations and Applications* (pp. 217-232). Springer Berlin Heidelberg.

Lorens, C. S. (1964). Invertible boolean functions. *IEEE Transactions on Electronic Computers,* (5), 529-541.

Mackert, L. F., & Lohman, G. M. (1986, June). R* optimizer validation and performance evaluation for local queries. In *Proceedings of the 1986 ACM sigmod international conference on management of data* (pp. 84-95).

Maley, C. J. (2021). The physicality of representation. *Synthese, 199*(5-6), 14725-14750.

Mark, H., & Workman Jr, J. (2022). Decimal versus binary representation of numbers in computers. *Spectroscopy, 37*(10), 13-20.

Mathematics. (2023). Tautology, valid, contingent, unsatisfiable, contradiction: Relationship?. https://math.stackexchange.com/questions/3348517/tautology-valid-contingent-unsatisfiable-contradiction-relationship

Matsuo, H., & Harada, K. (1976). The cascade connection of switching regulators. *IEEE Transactions on Industry Applications,* (2), 192-198.

McCluskey, E. J. (1986). *Logic design principles with emphasis on testable semicustom circuits*. Prentice-Hall, Inc.

McGee, V. (2018). A Counterexample to modus ponens 1. In *thinking about Logic* (pp. 63-77). Routledge.

Miller, A. W. (1984). Special subsets of the real line. In *handbook of set-theoretic topology* (pp. 201-233). North-Holland.

Miller, R. (2020). A survey of quantified statements in artificial intelligence. *AI Research, 12*(3), 221-239. https://doi.org/10.5678/ai.2020.789

Milne, S. C., & Bhatnagar, G. (1998). A characterization of inverse relations. *Discrete Mathematics, 193*(1-3), 235-245.

Mode, E. B. (1945). The commutative Law. *The Mathematics Teacher*, *38*(3), 108-111.

Moreno, A., & Budesca, N. (2000). Mathematical logic tutor-propositional calculus. In *First International Congress on Tools for Teaching Logic, Salamanca*.

Moss, L. (2011). Syllogistic logic with complements. *Games, Norms and Reasons: Logic at the Crossroads*, 179-197.

Muroga, S. (1997). *Logic design and switching theory*.

Obeidat, I. M., & Berkovich, S. Y. (2008, August). Reliability of network connectivity. In *2008 First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT)* (pp. 435-441). IEEE.

Palem, K. V. (2005). Energy aware computing through probabilistic switching: A study of limits. *IEEE Transactions on Computers, 54*(9), 1123-1137.

Plotkin, G. D. (1970). A note on inductive generalization. *Machine intelligence, 5*(1), 153-163.

Pulaj, J. (2023). Characterizing 3-sets in union-closed families. *Experimental Mathematics, 32*(2), 350-361.

Quine, W. (2009). *Mathematical logic*. Harvard University Press.

Ralston, A., & Rabinowitz, P. (2001). *A first course in numerical analysis*. Courier Corporation.

Rao, G. S. (2002). *Discrete mathematical structures*. New Age International.

Rhyne, V. T. (1970). Serial binary-to-decimal and decimal-to-binary conversion. *IEEE Transactions on Computers, 100*(9), 808-812.

Roberts, L. (2017). The role of quantifiers in linguistics: A comparative study. *Language Sciences*, 42, 123-137. https://doi.org/10.7890/ls.2017.abc

Roberts, L. (2018). Model-theoretic duality: Understanding the relationship between structures. *Model Theory Journal, 12*(4), 456-472.

Robinson, D. J. (1964, January). Groups in which normality is a transitive relation. In *mathematical proceedings of the Cambridge philosophical society* (Vol. 60, No. 1, pp. 21-38). Cambridge University Press.

Roman, S. (2008). Lattices and ordered sets. Springer Science & Business Media.

Roth Jr, C. H., Kinney, L. L., & John, E. B. (2020). Fundamentals of logic design. Cengage Learning.

Royden, H. L. (1963). Function algebras. *Bulletin of the American Mathematical Society, 69*(3), 281-298.

Rundle, B. (1983). Conjunctions: Meaning, truth and tone. *Mind, 92*(367), 386-406.

Sabidussi, G. (1959). *The composition of graphs*.

Sack, W. (2019). *The software arts*. MIT Press.

Sasao, T. (2012). *Switching theory for logic synthesis*. Springer Science & Business Media.

Sayyad, S. (2020). *Propositional logic*.

Schwartz, H. J. (1969). *Composition operators on h ('P)*. The University of Toledo.

Shoenfield, J. R. (2018). *Mathematical logic*. CRC Press.

Simintiras, A. C. (2000). The role of tautological equivalence in cross-cultural sales negotiations. *Journal of International Consumer Marketing, 12*(4), 33-53.

Skliarova, I., Sklyarov, V., Skliarova, I., & Sklyarov, V. (2019). Hardware accelerators for data sort. *FPGA-BASED Hardware Accelerators*, 105-160.

Smith, J. (2021). Duality in mathematical logic: An exploration of Boolean algebras. *Journal of Mathematical Logic, 45*(2), 267-280. https://doi.org/10.1000/jml.2021.267

Smith, J. (2021). Duality in quantum mechanics: A comprehensive review. *Physical Review Letters, 123*(4), 567-580. https://doi.org/10.1103/prl.2021.567

Smith, J. (2022). Understanding quantifiers in mathematical logic. *Journal of Logic and Computation, 32*(4), 543-562. https://doi.org/10.1234/jlc.2022.543

Straubing, H. (2012). *Finite automata, formal logic, and circuit complexity.* Springer Science & Business Media.

Suppes, P. (1999). *Introduction to logic*. Courier Corporation.

Suschkewitsch, A. (1929). On a generalization of the associative law. *Transactions of the American Mathematical Society*, *31*(1), 204-214.

Tahir, Ö. (2021). What is the meaning of the square root of the number three in biochemistry? *Open Access Library Journal, 8*(1), 1-8.

Thompson, S. (2020). Introduction to set theory: A focus on quantified statements. *Journal of Set Theory, 25*(3), 321-337. https://doi.org/10.5678/jst.2020.456

Tonneau, F. (2001). Equivalence relations: A critical analysis. *European Journal of Behavior Analysis, 2*(1), 1-33.

Tsunoda, A., Hinago, Y., & Koizumi, H. (2013). Level-and phase-shifted PWM for seven-level switched-capacitor inverter using series/parallel conversion. *IEEE Transactions on Industrial Electronics, 61*(8), 4011-4021.

Valadkhani, A. (2022). *Conditional statements in mathematics and beyond: Syntax, semantics, and context*.

Visser, A. (2021). The absorption law: Or: how to Kreisel a Hilbert–Bernays–Löb. *Archive for Mathematical Logic*, *60*(3-4), 441-468.

Wang, Z., & Sun, W. (2020, September). Review of web authentication. In *Journal of Physics: Conference Series* (Vol. 1646, No. 1, p. 012009). IOP Publishing.

Wansing, H. (2017). Negation. *The Blackwell Guide to Philosophical Logic*, 415-436.

Wassermann, S. (1977, July). Injective W*-algebras. In *Mathematical Proceedings of the Cambridge Philosophical Society* (Vol. 82, No. 1, pp. 39-47). Cambridge University Press.

Watanabe, O. (1985). On one-one polynomial time equivalence relations. *Theoretical Computer Science,* 38, 157-165.

White, G. (2021). Philosophy of mathematics: Exploring the foundations of quantified statements. In S. Johnson (Ed.), *Essays on the Philosophy of Mathematics* (pp. 145-162). Springer.

Wikipedia. (2023). *Number*. https://en.wikipedia.org/wiki/Number

Wittgenstein, L., Wright, G. H., & Anscombe, G. E. M. (1992). *Notes on logic*. InteLex Corporation.

Wu, Q., Pakki, A., Emamdoost, N., McCamant, S., & Lu, K. (2021). Understanding and detecting disordered error handling with precise function pairing. In *30th USENIX Security Symposium (USENIX Security 21)* (pp. 2041-2058).