3. Penalaran Matematika

Penalaran matematika (*mathematical reasoning*) diperlukan untuk menentukan apakah sebuah argumen matematika benar atau salah dan juga dipakai untuk membangun suatu argumen matematika. Penalaran matematika tidak hanya penting untuk melakukan pembuktian (*proof*) atau pemeriksaan program (*program verification*), tetapi juga untuk melakukan inferensi dalam suatu sistem kecerdasan buatan (*artificial intelligence*/AI).

Beberapa istilah yang akan dipakai dalam penalaran matematika perlu dimengerti artinya, yakni, , bukti, inferensi, teorema, *lemma*, *corollary* dan konjektur (*conjecture*). Aksioma (*axiom*) adalah asumsi dasar dari suatu struktur matematika yang tidak perlu bukti. Pembuktian (*proof*) dipakai untuk menunjukkan bahwa suatu pernyataan adalah benar. Suatu pembuktian terdiri dari rangkaian pernyataan-pernyataan yang membentuk sebuah argumen. Langkah-langkah yang menghubungkan pernyataan-pernyataan ini disebut sebagai aturan inferensi (*rules of inference*).

Suatu penalaran yang salah disebut sebagai *fallacy*. Teorema adalah pernyataan yang dapat ditunjukkan bernilai benar. Suatu *lemma* adalah teorema sederhana yang dipergunakan sebagai hasil-antara dalam pembuktian teorema lain, sedangkan *corollary* adalah suatu proposisi yang secara langsung diperoleh dari teorema yang sudah dibuktikan. Suatu konjektur adalah suatu pernyataan yang nilai kebenarannya tidak diketahui. Setelah pembuktian berhasil dilakukan, maka konjektur berubah menjadi teorema.

Aturan-aturan inferensi memberikan sarana untuk melakukan pembenaran dari langkah-langkah yang dipakai dalam proses pembuktian. Salah satu aturan penting yang perlu kita kenal adalah *modus ponens* atau *law of detachment*. Aturan ini didasarkan pada tautologi $(p \wedge (p \rightarrow q)) \rightarrow q$. Kita menuliskan modus ponen dengan cara berikut:

$$\begin{array}{l} p \\ \underline{p \rightarrow q} \\ \therefore q \end{array}$$

Dua buah hipotesis, p dan $p \rightarrow q$, dituliskan dalam sebuah kolom, dan kesimpulan (*conclusion*) dibawah suatu baris, sedangkan tanda "$\therefore$" dibaca sebagai "jadi " atau "oleh karena itu". Lebih lanjut lagi, bentuk umum dari aturan inferensi adalah

$$\begin{array}{l} p_1 \\ p_2 \\ \cdots \\ \underline{p_n} \\ \therefore q \end{array}$$

Aturan ini menyatakan bahwa, jika p_1 dan p_2 dan … dan p_n semuanya benar, maka q juga akan benar. Aturan inferensi ini dapat dipakai dalam argumen matematika tanpa memerlukan pembuktian apapun.

Disamping aturan inferensi, dikenal juga argumen yang juga terdiri dari satu atau beberapa buah hipotesis dan suatu kesimpulan. Suatu argumen disebut *valid*, apabila, saat semua hipotesisnya benar, maka kesimpulannya juga benar. Tetapi, jika suatu hipotesis salah, argumen yang valid sekalipun dapat mengakibatkan kesimpulan yang juga salah, seperti ditunjukkan pada contoh berikut. Contoh: Pehatikan pernyataan berikut ini :

"Jika n dapat dibagi dengan 3, maka n^2 juga dapat dibagi 9.
"101 dapat dibagi 3. Jadi, 101^2 dapat dibagi 9."

Berdasarkan *modus ponens*, argumen diatas *valid*. Akan tetapi kesimpulannya salah, karena satu dari hipotesisnya salah (yakni "101 dapat dibagi 3."). Jika dalam argumen diatas kita gantikan 101 dengan 102, maka kita akan secara benar menyimpulkan bahwa 102^2 dapat dibagi dengan 9.

Suatu teorema dapat dibuktikan dengan cara langsung maupun secara tidak langsung. Dalam pembuktian langsung, suatu implikasi $p \rightarrow q$ dapat dibuktikan dengan menunjukkan bahwa jika p benar, maka q juga benar. Contoh: berikan pembuktian langsung teorema berikut ini:

"Jika n ganjil, maka n^2 juga ganjil."

<u>Ide</u>: asumsikan bahwa hipotesis dari implikasi ini benar (yakni, n ganjil). Lalu gunakan aturan inferensi dan teorema yang telah diketahui untuk menunjukkan bahwa q juga benar (yakni n^2 ganjil).

Bukti: asumsikan n ganjil, maka n bisa dinyatakan sebagai $n = 2k + 1$, dimana k bilangan bulat. Akibatnya,

$$n^2 = (2k + 1)^2 = 4k^2 + 4k + 1 = 2(2k^2 + 2k) + 1$$
$$= 2m + 1; \quad \text{dimana } m = 2k^2 + 2k$$

Karena n^2 dapat dituliskan dalam bentuk tersebut diatas, maka n^2 adalah juga bilangan ganjil.

Suatu implikasi $p \rightarrow q$ adalah ekivalen dengan bentuk *contra-positive* nya, yakni $\neg q \rightarrow \neg p$. Oleh karena itu, dalam pembuktian tidak langsung, implikasi $p \rightarrow q$ dapat dibuktikan dengan menunjukkan bahwa, saat q salah, maka p juga salah. Contoh: berikan bukti tak langsung teorema

"Jika $3n + 2$ ganjil, maka n adalah ganjil."

Ide: asumsikan bahwa kesimpulan dari implikasi ini salah (n genap). Kemudian gunakan aturan inferensi dan teorema yg telah diketahui untuk menunjukkan bahwa p juga salah ($3n+2$ genap).

Bukti : Tinjau n genap, sehingga bisa dinyatakan sebagai $n = 2k$, dimana k bilangan bulat. Dengan demikian

$$3n + 2 = 3(2k) + 2 = 6k + 2 = 2(3k + 1)$$
$$= 2m, \quad \text{dimana } m = 3k + 1$$

Oleh karena itu, $3n + 2$ adalah bilangan genap. Dengan demikian, kita telah menunjukkan bahwa kontrapositif dari implikasi bernilai benar. Dengan demiikian implikasinya sendiri juga benar (yakni, "jika $2n+3$ ganjil, maka n ganjil").

Dalam pembuktian, ada suatu teknik penting yang disebut sebagai prinsip induksi, yang merupakan suatu cara (*tool*) untuk membuktikan bahwa predikat tertentu bernilai benar untuk semua bilangan cacah. Prinsip ini tidak dapat dipakai untuk menemukan suatu teorema, melainkan hanya untuk membuktikannya saja.

Jika kita punya fungsi proposisi P(n), dan kita ingin membuktikan bahwa P(n) benar untuk sebarang bilangan cacah n, kita lakukan langkah-langkah berikut:

- Menunjukkan bahwa P(0) benar (langkah dasar)
- Menunjukkan bahwa jika P(n) benar maka P(n+1) juga benar untuk sebarang $n \in$N. (langkah induktif)
- Maka P(n) haruslah benar untuk sebarang $n \in$N. (kesimpulan)

Contoh: Tunjukkan bahwa $n < 2^n$ untuk semua bilangan bulat positif n.

Jawab: Misalkan P(n) adalah proposisi "$n < 2^n$"

1. Tunjukkan bahwa P(1) benar. (langkah dasar)

   Untuk n=1, diperoleh relasi $1<2^1 = 2$. Jadi P(1) benar

2. Tunjukkan bahwa, jika P(n) benar, maka P(n + 1) juga benar.(langkah induktif)

   Asumsikan $n < 2^n$ benar.

   Kita perlu menunjukkan bahwa P(n + 1) adalah benar, yaitu $n + 1 < 2^{n+1}$ benar.

   Kita mulai dari $n < 2^n$:

   $n+1 < 2^n+1 \leq 2^n + 2^n = 2^{n+1}$

   Oleh karena itu, jika $n < 2^n$ maka n c+ $1 < 2^{n+1}$

3. Maka P(n) haruslah benar untuk sebarang bilangan bulat positif. (kesimpulan)

   $n< 2^n$ adalah benar untuk sebarang bilangan bulat positif.

   Akhir dari pembuktian.

Contoh lain : Buktikan formula "Gauss" $1 + 2 + \ldots + n = n\,(n + 1)/2$

Jawab :

1. Tunjukkan bahwa P(0) adalah benar. (langkah dasar)

   Untuk n = 0 kita peroleh 0 = 0. Jadi P(0) Benar.

2. Tunjukkan bahwa, jika P(n) maka P(n+1) untuk sebarang $n \in$N. (langkah induktif)

$1 + 2 + \ldots + n = n\,(n + 1)/2$

$$\begin{aligned}1 + 2 + \ldots + n + (n + 1) &= n\,(n + 1)/2 + (n + 1)\\ &= (2n + 2 + n\,(n + 1))/2\\ &= (2n + 2 + n^2 + n)/2\\ &= (2 + 3n + n^2)/2\\ &= (n + 1)\,(n + 2)/2\\ &= (n + 1)\,((n + 1) + 1)/2\end{aligned}$$

3. Maka P(n) haruslah benar untuk sebarang $n \in$N. (kesimpulan)

   $1 + 2 + \ldots + n = n\,(n + 1)/2$ adalah benar untuk semua $n \in$N.

   Akhir dari pembuktian.

Disamping prinsip induksi matematika yang telah dijelaskan, ada teknik pembuktian lain yang sangat mirip dengan prinsip induksi matematika yang disebut sebagai *prinsip kedua dari induksi matematika*. Prinsip ini dapat dipergunakan untuk membuktikan bahwa suatu fungsi proposisi P(n) bernilai benar untuk sebarang bilangan cacah n. Langkah-langkah pembuktian dalam prinsip kedua adalah sebagai berikut:

- Tunjukkan bahwa P(0) adalah benar. (langkah dasar)
- Tunjukkan bahwa jika P(0) dan P(1) dan … dan P(n), maka P(n+1) benar untuk sebarang $n \in$N. (langkah induktif)
- Maka P(n) haruslah benar untuk sebarang $n \in$N. (kesimpulan)

Contoh: tunjukkan bahwa setiap bilangan bulat yang lebih besar dari 1 dapat dituliskan sebagai hasil perkalian bilangan-bilangan prima.

- Tunjukkan bahwa P(2) adalah benar. (langkah dasar)

  2 adalah hasil perkalian satu buah bilangan prima, yaitu dirinya sendiri.

- Tunjukkan bahwa jika P(2) dan P(3) dan … dan P(n) benar, maka P(n+1) benar untuk sebarang $n \in$N. (langkah induktif)

  Disini ada dua kemungkinan:

  - Jika (n+1) bilangan prima, maka jelas bahwa P(n +1) benar.

- Jika (n+1) adalah komposit, maka bilangan tersebut akan dapat dituliskan sebagai hasil perkalian dua bilangan bulat a dan b sedemikian hingga

  $$2 \leq a \leq b < n + 1.$$

  Berdasarkan hipotesa induksi, baik a maupun b dapat dituliskan sebagai hasil perkalian bilangan-bilangan prima. Jadi, $n + 1 = a \cdot b$ dapat dituliskan sebagai hasil perkalian bilangan-bilangan prima.

- Dengan demikian, P(n) haruslah benar untuk sebarang $n \in$N. (kesimpulan)
  Akhir dari pembuktian.

  Kita telah menunjukkan bahwa sebarang bilangan bulat yang lebih besar dari 1 dapat dituliskan sebagai hasil perkalian bilangan-bilangan prima.

Salah satu prinsip lain yang juga berkaitan dengan induksi matematika, sekaligus dengan algoritma adalah prinsip *rekursi*, dimana suatu objek didefinisikan dari dirinya sendiri. Kita dapat mendefinisikan deretan, fungsi dan himpunan secara rekursif. Perhatikan contoh berikut.

Contoh: suatu deretan $\{a_n\}$ dinyatakan sebagai $a_n = 2^n$ untuk n = 0, 1, 2, … . Secara rekursif, deretan tersebut dapat juga didefinisikan sebagai

$$a_0 = 1$$
$$a_{n+1} = 2a_n \quad \text{untuk } n = 0, 1, 2, \ldots$$

Jelaslah bahwa induksi dan rekursi adalah prinsip yang sama.

Kita dapat menggunakan metoda berikut ini untuk mendefinisikan suatu fungsi dengan domain bilangan cacah:

1. Tentukan nilai fungsi pada (argumen) nol.
2. Berikan aturan untuk mencari nilainya pada sebarang bilangan bulat berdasarkan nilainya pada bilangan bulat yang lebih kecil.

Definisi yang demikian disebut sebagai definisi rekursif atau definisi induktif. Perhatikan contoh berikut ini.

- Suatu fungsi didefinisikan secara rekursif dengan formula berikut $f(0) = 3$, $f(n+1)=2f(n) + 3$, maka nilai fungsi tsb untuk beberapa harga n adalah

$$f(0) = 3$$
$$f(1) = 2f(0) + 3 = 2\times 3 + 3 = 9$$
$$f(2) = 2f(1) + 3 = 2\times 9 + 3 = 21$$
$$f(3) = 2f(2) + 3 = 2\times 21 + 3 = 45$$
$$f(4) = 2f(3) + 3 = 2\times 45 + 3 = 93$$

- Definisikan fungsi *faktorial* $f(n) = n!$ secara rekursif!
  Jawab: definisi rekursif fungsi faktorial adalah $f(0) = 1$, $f(n+1) = (n+1)f(n)$
  Evaluasi untuk beberapa harga n menghasilkan:

$$f(0) = 1$$
$$f(1) = 1\times f(0) = 1\cdot 1 = 1$$
$$f(2) = 2\times f(1) = 2\cdot 1 = 2$$
$$f(3) = 3\times f(2) = 3\cdot 2 = 6$$
$$f(4) = 4\times f(3) = 4\cdot 6 = 24$$

- Definisikan barisan Fibonacci secara rekursif!
  Jawab: definisi rekursif dari barisan Fibonacci adalah:

  $f(0) = 0$, $f(1) = 1$, dan $f(n) = \mathrm{f}(n - 1) + f(n - 2)$

  Evaluasi untuk beberapa harga n menghasilkan

$$f(0) = 0$$
$$f(1) = 1$$
$$f(2) = f(1) + f(0) = 1 + 0 = 1$$

$f(3) = f(2) + f(1) = 1 + 1 = 2$

$f(4) = f(3) + f(2) = 2 + 1 = 3$

$f(5) = f(4) + f(3) = 3 + 2 = 5$

$f(6) = f(5) + f(4) = 5 + 3 = 8$

Untuk mendefinisikan suatu himpunan secara rekursif, kita memerlukan dua hal berikut:

- sekumpulan elemen-elemen awal,
- aturan untuk mendefinisikan elemen-elemen tambahan dari elemen yang sudah ada didalam himpunan.

Contoh: Misalkan S didefinisikan secara rekursif sbb:

$3 \in S$

$(x + y) \in S$ jika $(x \in S)$ dan $(y \in S)$

Maka, S adalah himpunan dari bilangan bulat positif yang dapat dibagi 3.

Bukti: misalkan A himpunan semua bilangan bulat positif yang dapat dibagi 3. Untuk menunjukkan A = S, kita harus membuktikan bahwa $A \subseteq S$ and $S \subseteq A$.

Bag. I: Untuk membuktikan bahwa $A \subseteq S$, kita harus menunjukkan bahwa setiap bilangan bulat positif yang dapat dibagi 3 terkandung didalam S. Untuk menunjukkan hal ini kita akan memakai induksi matematika. Misalkan P(n) adalah pernyataan "3n anggota S".

Langkah dasar. P(1) benar, karena $3 \in S$.

Langkah induktif. Untuk menunjukkan: Jika P(n) benar, maka P(n + 1) benar.
Asumsikan 3n anggota S. Karena 3n anggota S dan 3 adalah anggota S, dari definisi rekursif S maka $3n + 3 = 3(n + 1)$ adalah juga anggota S. Kesimpulan Bag.-an I: $A \subseteq S$.

Bag. II: Akan ditunjukkan: $S \subseteq A$.

Langkah dasar. Akan ditunjukkan bahwa semua elemen awal dari S adalah juga anggota A.

3 anggota A. Benar.

Langkah induktif. Akan ditunjukkan:($x + y$) adalah anggota A manakala x dan y anggota S.

Jika x dan y keduanya anggota A, maka 3|x dan 3|y. Seperti telah kita ketahui, ini berarti 3|(x+y).

Kesimpulan Bagian II: S $\subseteq$ A.
Kesimpulan Keseluruhan: A = S.

Berikut ini kita tinjau contoh lain. Suatu formula yang *well-formed* dari variabel, bilangan dan operator {+, -, *, /, ^} didefinisikan sbb:

- x adalah suatu formula yang *well-formed* jika x merupakan suatu bilangan atau variabel.
- $(f + g)$, $(f - g)$, $(f * g)$, (f / g), $(f \wedge g)$ adalah formula yang *well-formed* jika f dan g juga formula yang *well-formed*.

Dengan definisi ini, kita dapat membentuk formula-formula berikut ini:

$(x - y)$
$((z / 3) - y)$
$((z / 3) - (6 + 5))$
$((z / (2 * 4)) - (6 + 5))$

Disamping fungsi, ada sekelompok algoritma tertentu yang juga bersifat rekursif. Suatu algoritma disebut rekursif jika algoritma tersebut memecahkan suatu permasalahan dengan cara mereduksi permasalahan itu menjadi permasalahan yang sama dengan masukan yang lebih kecil. Salah satu contoh dari algoritma ini adalah algoritma Euklidian rekursif, yang pseudocodenya diperlihatkan di bawah ini.

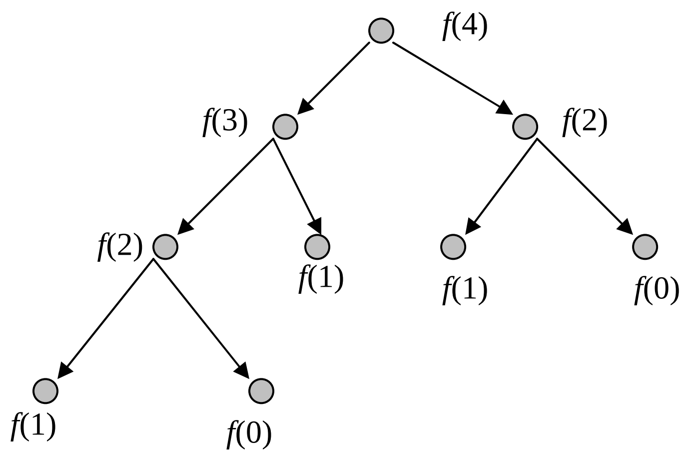
```
procedure gcd(a,b: bilangan bulat tak-negatif dng a<b)
     if a=0 then gcd(a,b):=b
     else gcd(a,b) := gcd(b mod a,a)
```

Didalam algoritma *gcd* rekursif tersebut, *procedure gcd* memanggil dirinya sendiri. Ini adalah cirri khas dari algoritma rekursif.

Disamping itu, algoritma Fibonacci yang telah dibahas pada bab terdahulu juga dapat dituliskan secara rekursif sebagai berikut.

```
procedure fibo(n: nonnegative integer)
  if n = 0 then fibo(0):= 0
    else if n = 1 then fibo(1):= 1
      else fibo(n):= fibo(n–1)+fibo(n–2)
```

Untuk suatu bilangan *n* tertentu, algoritma ini menjalankan rekursi dan mereduksi nilai *n* sampai ke nilai yang terdefinisi, yaitu *n*=1 dan *n*=0. Selanjutnya, evaluasi nilai *n* yang lebih tinggi bisa dilakukan. Proses eksekusi algoiritma dengan *n*=4 diperlihatkan pada diagram pohon berikut ini.



Sebagai perbandingan, berikut ini diperlihatkan *pseudocode* dari algoritma Fibonnaci iteratif.

```
procedure iterative_fibo(n: nonnegative integer)
if n = 0 then y := 0
else
begin
     x := 0
     y := 1
     for i := 1 to n-1
     begin
```

```
            z := x + y
            x : = y
            y := z
        end
    end {y adalah bilangan Fibonacci ke-n}
```

Untuk setiap algoritma rekursif, terdapat algoritma iteratif yang ekivalen. Algoritma rekursif sering kali lebih singkat, lebih elegan, dan lebih mudah dipahami dibandingkan algoritma iteratif-nya. Tetapi, algoritma iteratif biasanya lebih efisien menurut ukuran kompleksitas ruang dan waktu.