MODUL 1: PENGANTAR TEORI BAHASA

PENGANTAR

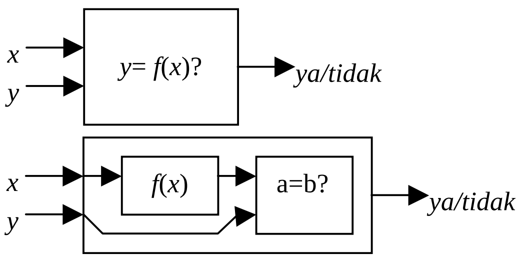
Kuliah ini membahas model-model komputasi sebagai mesin abstraks yang dapat didefinisikan secara matematis, mulai dari yang paling sederhana hingga yang paling powerful.

Model-model sederhana dibahas agar formalisasi matematis dapat terbentuk secara bertahap, selain itu tetap masih ada hubungannya dengan situasi-situasi dunia nyata (masih ada maslah-masalah yang sesuai dengan model-model komputasi sedehana tsb).

Secara umum model-model digambarkan sebagai mesin untuk menjawab masalah keputusan berdasarkan masukan string *x* dengan memberikan keluaran berupa: **ya** atau **tidak** misalnya:

- apakah *x* adalah bilangan prima?
- apakah suatu string s anggota dari bahasa B?

➢ Masalah komputasi memang lebih umum daripada masalah keputusan, namun pada dasarnya suatu model untuk masalah keputusan memerlukan komponen yang dapat melakukan komputasi yang terkait, misalnya:
  - untuk suatu (*x*, *y*), "apakah $y = f(x)$?" hanya dapat dijawab jika $f(x)$ dapat dikomputasi.



Model-model mesin yang akan dibahas: Finite Automata (FA), Pushdown Automata (PDA), dan Mesin Turing (TM). Dan, bahasa-bahasa yang terkait dengan masing-masing mesin di atas: bahasa regular, bahasa context-free, bahasa rekursif, dst.

FA:
- merupakan model dimana setiap saat berada dalam suatu status di antara sejumlah terbatas status-status diskretnya, serta akan berpindah dari satu status ke status lain dengan cara yang dapat terprediksi sebagai respon terhadap setiap simbol masukan tunggal.
- dapat digunakan untuk mengenal bahasa-bahasa regular.
- keterbatasan: tidak memiliki memori kecuali pada mesin bisa didefinisikan sejumlah status dari mesin dengan jumlah berhingga.

PDA:
- Merupakan FA yang dilengkapi suatu memory berstruktur stack tunggal namun dengan kapasitas tak berhingga
- Dengan stack tersebut dapat mengenali bahasa context-free yaitu suatu kelas bahasa yang lebih tinggi dari bahasa regular.

TM:
- Adanya sequential storage sebagai memory yang dapat di baca-tulis.
- Dapat mengenali kelas bahasa yang lebih tinggi dari bahasa context-free
- Bukan saja dapat menjawab "ya" dan "tidak" pada setiap string masukan, tetapi tape menjadi medium keluaran proses
- dalam teori komputasi akan menjadi alat ukur untuk membandingkan kompleksitas satu masalah komputasi dengan masalah lain

Kelas-kelas bahasa merepresentasikan kelas-kelas masalah dan model-model mesin abstraks tersebut merepresentasikan kelas metoda pemecah masalah.

Teori dan model komputasi telah berkembang jauh sebelum perkembangan perangkat komputer itu sendiri.

Notasi-notasi dan konsep-konsep penting yang perlu diingat kembali karena digunakan di dalam kuliah ini adalah sbb.

Mengenai Teori Himpunan:

- Penulisan himpunan $A = \{00, 01, 10. 11\} = \{ab \mid a, b$ berharga 0 atau 1$\}$
- $\in$: anggota himpunan
- $\notin$: bukan anggota himpunan
- U: himpunan universal (semua kemungkinan elemen)
- $\emptyset$: himpunan kosong (himpunan tanpa elemen)
- Himpunan komplemen $A' = \{x \in U \mid x \notin A\}$
- Diagram Venn: memperlihatkan hubungan/operasi antara himpunan
- Opr. gabungan: $A \cup B = \{x \mid x \in A$ atau $x \in B\}$
- Opr. irisan: $A \cap B = \{x \mid x \in A$ dan $x \in B\}$
- Opr. perbedaan: $A - B = \{x \mid x \in A$ dan $x \notin B\}$
- Opr. perbedaan simetris: $A \oplus B = (A - B) \cup (B - A)$
- "A dan B disjoin": jika $A \cap B = \emptyset$,
- himpunan bagian: "A $\subseteq$ B" jika untuk setiap $x \in A$ maka $x \in B$ tetapi ada kemungkinan $y \in B$, ternyata $y \notin A$.
- himpunan bagian: "A $\subset$ B" jika untuk setiap $x \in A$ maka $x \in B$ tetapi pasti ada kemungkinan $y \in B$, ternyata $y \notin A$.
- power set (2^A): himpunan dari semua himpunan bagian yang mungkin dapat dibuat dari himpunan A
- pasangan terurut (a, b)
- Cartesian product ($A \times B$): $\{(a, b) \mid a \in A, b \in B\}$
- partisi himpunan A menjadi $A_1, A_2, \ldots A_n$: A_i dan A_j disjoint (untuk $i \neq j$) dan $A = A_1 \cup A_2 \cup \ldots \cup A_n$.
- Sifat-sifat: komutatif, asosiatif, distributif, idempoten, absorptif
- hukum De Morgan: $A \cup B = (A' \cap B')'$

Mengenai Fungsi (f: $A \to B$):

- fungsi parsial
- fungsi total
- fungsi injeksi (satu-ke-satu)
- fungsi surjeksi (onto)
- fungsi bijeksi
- Komposisi fungsi-fungsi ($g \circ f$)
- invers dari fungsi ($x = f^{-1}(y)$)
- Operasi uner ($u(x)$)
- operasi biner ($f(x, y)$)

Mengenai Relasi ($a\ R\ b$)

- Sifat-sifat relasi: refleksif, simetri, transitivitas
- relasi ekivalen
- Kelas ekivalen berisikan a ($[a]_R$)

KONSEP MENGENAI BAHASA

Simbol. Simbol adalah elemen unik terkecil dari bahasa. Dari suatu bahasa terdapat sejumlah berhingga simbol yang digunakan.

Alfabet. Kita sebut himpunan berhingga simbol suatu bahasa sebagai alfabet simbol (atau disingkat alfabet) dari bahasa tersebut.

String. Simbol-simbol menyusun string simbol (atau disingkat string). Dalam suatu string bisa ada suatu simbol yang muncul lebih dari satu kali dan ada juga simbol yang tidak digunakan.

Bahasa. Bahasa merupakan himpunan dari sejumlah berhingga atau tidak berhingga string-string dari alfabet bahasa tersebut. Jadi, suatu bahasa merupakan himpunan bagian himpunan seluruh kemungkinan string yang dapatdibentuk dari alfabet bahasa tersebut.

Grammar. Susunan simbol-simbol dalam string-string suatu bahasa mengikuti aturan-aturan grammar (pembentukan) tertentu. Kompleksitas aturan-aturan pembentukan ini yang akan membedakan kelas-kelas bahasa yang akan dibahas nanti.

Definisi di atas walaupun nampaknya amat menyederhanakan pemahaman kita sehari-hari akan bahasa manusia, sebenarnya cukup mewakili hingga tingkat gramatikal. Secara gramatikal bahasa manusia, misalnya bahasa Inggris, adalah himpunan semua kalimat di mana setiap kalimat tersusun atas kata-kata dengan aturan gramatikal yang tertentu. Suatu bahasa merupakan himpunan kalimat dan setiap kalimat terbentuk oleh huruf-huruf. Jadi alfabet dalam hal ini adalah alfabet dalam pengertian yang umumnya. Dari sudut pandang yang lain bisa juga suatu kata dapat diasosiasikan sebagai suatu simbol unik terkecil, dan satu kalimat adalah string dari kata-kata sementara himpunan kata-kata atau kosakata dapat dipandang sebagai alfabet. Atau bisa juga dalam representasi biner ASCII alfabetnya = {0, 1} yang nama setiap simbol ASCII direpresentasikan dengan delapan bit biner.

Dalam pengertian bahasa yang lain, misalnya pada bahasa pemrograman, sebuah program lengkap adalah suatu string. Keyword-keyword, nama-nama variabel, nama-nama fungsi, tanda-tanda operasi dan notasi adalah simbol-simbol di dalam bahasa ini.

Suatu string yang menjadi anggota suatu bahasa sering juga kita sebut “kata (word)”. Dengan demikian istilah “kata” juga memiliki arti yang lebih umum dari pengertian “kata” dalam bahasa sehari-hari.

Notasi Alfabet. Untuk memudahkan penulisan notasi selanjutnya maka himpunan simbol atau alfabet kita tulis dengan lambang Σ. Dalam model bahasa yang akan dibahas untuk memudahkan pembahasannya digunakan himpunan simbol yang hanya berisi dua bahkan mungkin satu simbol. Namun secara umum konsep-konsep yang dipelajari akan berlaku untuk himpunan simbol yang lebih besar. Contoh bahasa-bahasa yang terbentuk dari $\Sigma = \{a, b\}$:

$\{\Lambda, a, aa, ab\}$

$\{x \in \{a, b\}^* \mid |x| \leq 8\}$

$\{x \in \{a, b\}^* \mid |x| \in \text{bilangan ganjil}\}$

Notasi Variabel Simbol/string. Dalam pembahasan, suatu variabel string tertentu sering direpresentasikan sebagai *r*, *s*, *t*, .., *x*, *y*, atau *z*. (dengan mengambil huruf kecil di urutan belakang dalam abjad), sementara suatu variabel simbol sering direpresentasikan sebagai *a*, *b*, *c*,… yaitu huruf-huruf kecil pada urutan terdepan di dalam abjad.

Notasi variabel himpunan string. Variabel untuk himpunan string dinotasikan dengan huruf kapital pada urutan awal abjad. $L = \{001, 1110, 01, 000001\}$.

Panjang String. Suatu string tersusun atas sejumlah n simbol, $n \geq 0$. Panjang string x ditulis $|x|$.

String Kosong, Jika $n = 0$ maka string tersebut disebut string kosong**.** Untuk konvensi selanjutnya dalam pembahasan ini suatu string kosong dinotasikan notasi Λ. **Note**: Himpunan yang berisi hanya string kosong bukanlah himpunan kosong ($\{\Lambda\} \neq \varnothing$, tetapi $\{\} = \varnothing$).

Contoh: $\Sigma = \{a, b\}$, contoh-contoh string yang bisa dibentuk dari Σ adalah Λ, *a*, *abaa*, dan *aabbaa*. **Note**: *a* dalam hal ini maka bisa berarti salah satu simbol dari $\Sigma = \{a, b\}$, atau string dengan panjng 1. $|aaa| = |aba| = 3$. $|a| = 1$. $|\Lambda| = 0$.

Operasi konkatenasi pada string-string. String x dan y apabila dikonkatenasi akan menjadi string xy. Contoh: Jika $x = aaba$ dan $y = bba$ maka $xy = aababba$. Jika $x = aaba$ dan $y = \Lambda$, maka $xy = x$.

Operasi konkatenasi berulang pada string. Operasi konkatenasi dapat dilakukan sekian kali pada string yang sama. Untuk menyingkatkan penulisan konkatenasi berulang ini pada suatu string dapat dituliskan dengan notasi pangkat. contohnya $x^4 = xxxx = aabaaabaaabaaaba$. (**Ingat**: dalam hal ini notasi pangkat pada operasi string berbeda dengan notasi pangkat dalam operasi aritmetika). Secara umum konkatenasi adalah x^n dengan $n \geq 0$, yang mana jika $x^0 = \Lambda$.

Operasi konkatenasi pada himpunan string. Jika L_1 dan L_2 himpunan-himpunan string maka L_1L_2 adalah himpunan string $\{xy \mid x \in L_1 \text{ dan } y \in L_2\}$
Jika $L_1 = \{aa, bab\}$ dan $L_2 = \{bb, a\}$, maka $L_1L_2 = \{aabb, aaa, babbb, baba\}$. Jelaslah bahwa $L_1L_2 \neq L_2L_1$. Tapintuk $L_1 = \{\Lambda\}$ maka $L_1L_2 = L_2L_1 = L_2$.

Operasi konkatenasi berulang pada alfabet. Operasi konkatenasi dapat dilakukan sekian kali pada alfabet Σ. Untuk menyingkatkan penulisan konkatenasi berulang ini dapat dituliskan dengan notasi pangkat. Contohnya untuk $\Sigma = \{a, b\}$, $\Sigma^3 = \Sigma\Sigma\Sigma = \{aaa, aab, aba, abb, baa, bab, bba, bbb\}$. Secara umum konkatenasi adalah Σ^n dengan $n \geq 0$, yang mana jika $\Sigma^0 = \{\Lambda\}$.

Operasi konkatenasi berulang pada himpunan string. Operasi konkatenasi dapat dilakukan sekian kali pada himpunan string yang sama. Untuk menyingkatkan penulisan konkatenasi berulang ini dapat dituliskan dengan notasi pangkat. Contohnya untuk $L = \{aa, b\}$, $L^3 = LLL = \{aaaaaa, aaaab, aabaa, aabb, baaaa, baab, bbaa, bbb\}$. Secara umum konkatenasi adalah L^n dengan $n \geq 0$, yang mana jika $L^0 = \{\Lambda\}$.

Jadi, untuk $a \in \Sigma$, $x \in L$, dan $L \subseteq \Sigma^*$, maka:

$a^k = aa\ldots a$
$x^k = xx\ldots x$
$\Sigma^k = \Sigma\Sigma\ldots\Sigma = \{x \in \Sigma^* \mid |x| = k\}$
$L^k = LL\ldots L$
$a^0 = \Lambda$, $x^0 = \Lambda$, $\Sigma^0 = \{\Lambda\}$, $L^0 = \{\Lambda\}$
$\Sigma = \{0, 1\}$, maka $\Sigma\Sigma = \{00, 01, 10, 11\}$
$L = \{0, 111\}$, maka $LLL = \{000, 00111, 01110, 0111111, 11100, 1110111, 1111110, 111111\}$

Operasi Kleene-* pada Alfabet. Untuk alfabet Σ maka himpunan seluruh string yang mungkin terbentuk dari Σ adalah Σ^* (Kleene-* dari Σ). Dan, setiap bahasa L yang menggunakan alfabet Σ adalah subset dari Σ^*.

Contoh: Jika $\Sigma = \{a, b\}$, maka $\Sigma^* = \{\Lambda, a, b, aa, ab, ba, bb, aaa, aab, aba, baa, abb, bab, bba, bbb, \ldots\}$.

Operasi Kleene-* pada himpunan string. Bila L^k adalah himpunan semua string yang terbentuk dengan konkatenasi k elemen dari L maka L^* adalah himpunan semua string dari berbagai konkatenasi yang bisa dilakukan pada setiap elemen dari L. Ditulis

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

Operasi ini disebut Kleene-* (mengambil nama S.C. Kleene). L^* termasuk Λ karena $L^0 = \{\Lambda\}$.

Operasi Kleene-$^+$ pada himpunan string.
Untuk meniadakan L^0 maka digunakan notasi L^+ yang didefinisikan sebagai operasi Kleen-$^+$ sbb:

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$

Dengan demikian $L^+ = LL^* = L^*L$. Bukti: menurut definisinya $L^+ = L^1 \cup L^2 \cup L^3 \cup L^4 \cup \ldots = LL^0 \cup LL^1 \cup LL^2 \cup LL^3 \cup \ldots = L(L^0 \cup L^1 \cup L^2 \cup L^3 \cup \ldots) = LL^*$

Operasi-operasi Himpunan Umum
Karena bahasa-bahasa merupakan himpunan-himpunan string maka bahasa-bahas baru dapat dibentuk sebagai hasil operasi himpunan: gabungan, irisan, perbedaan, kompelemen. Operasi-operasi ini dapat dilakukan baik pada bahasa-bahasa dari alfabet yang sama maupun yang berbeda.

Contoh:

$L_1 \subseteq \Sigma_1^*$ dan $L_2 \subseteq \Sigma_2^*$ maka $(L_1 \cup L_2)$ adalah subset dari $(\Sigma_1 \cup \Sigma_2)^*$.

Operasi-operasi string lain
- x substring dari string y jika terdapat string lain w dan z sehingga $y = wxz$.
- x prefiks dari string y jika terdapat string lain z sehingga $y = xz$.
- x sufiks dari string y jika terdapat string lain w sehingga $y = wx$.

Bahasa Paling Sederhana dari Σ. Himpunan string subset dari Σ^* yang berisi string tunggal dengan panjang 0 atau 1 disebut bahasa paling sederhana dari Σ. Contoh $\Sigma = \{a, b, c\}$, maka bahasa-bahasa paling sederhana dari Σ adalah $\{\Lambda\}$, $\{a\}$, $\{b\}$, dan $\{c\}$.