String & Bahasa

Kuliah Teori Bahasa dan Automata
Program Studi Ilmu Komputer
Fasilkom UI

Prepared by:
Suryana Setiawan

Revised by:

Maya Retno Ayu S.

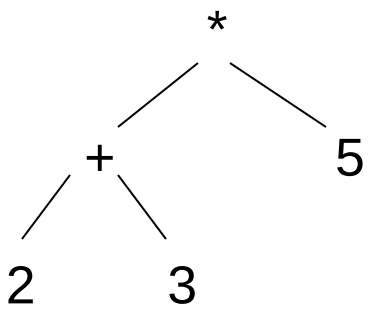
Pengantar

- Bagaimana komputer menyelesaikan problem ini?

```
int a, b;
a = 5;
b = (2 + 3) * 5;
```

Lexical Analysis

Parsing (Tree)



Optimization

Termination

Pada kuliah ini kita akan menggunakan suatu framework untuk Menyelesaikan berbagai problem yang beragam, yaitu:

LANGUAGE RECOGNITION

Pengantar

- Bahasa merupakan pokok bahasan utama dalam kuliah ini sehingga terminologi dasar terkait **bahasa** akan disampaikan.
- Suatu bahasa sebagai suatu himpunan dapat dipandang sebagai hasil operasi himpunan antara sejumlah bahasa lainnya, oleh sebab itu sejumlah **notasi operator** akan juga disampaikan.
- Bahasa sendiri sebagai himpunan berisikan sejumlah berhingga/tak berhingga string dari simbol-simbol yang didefinisikan dalam alfabet bahasa tersebut.
  - Terminologi hal-hal ini dijelaskan pada slide berikut

Terminologi Dasar

- **Simbol**: elemen pembentuk string yang biasanya menggunakan simbol-simbol alfanumerik (dalam buku teks disebut juga dengan nama **karakter**).
- **Alfabet simbol (notasi Σ)**: himpunan berhingga simbol-simbol pembentuk string.
- **String**: deretan simbol dari beberapa alfabet **Σ** sebagai satu kesatuan arti (suatu simbol bisa muncul sekali, berulang atau 0 kali)
- Σ^* : semua kemungkinan string yang dapat dibentuk dari alfabet Σ tersebut, suatu himpunan semesta.
- **Bahasa**: himpunan string dengan spesifikasi tertentu, jadi jika Σ^* adalah himpunan semesta dari string, maka suatu bahasa adalah subset di dalamnya.

Bahasa Kosong dan String Kosong

- **String kosong (*empty string dengan* notasi ε)**: suatu string tanpa berisi simbol satupun

- **Bahasa kosong (*empty language*)** : suatu bahasa yang tidak memiliki string satupun dan seperti halnya himpunan kosong, dinotasikan dengan ∅ or {}.

➔ Perhatian bahwa {ε} ≠ ∅, karena {ε} tidak kosong (berisi satu string ε, yaitu string kosong)!!!

Alfabet-alfabet sederhana

- Tanpa kehilangan sifat umumnya, dalam pembahasan kita selanjutnya, kita cenderung menggunakan alfabet-alfabet sederhana sbb:
  - Alfabet {1} untuk membentuk string-string 11, 111111, 11,…
  - Alfabet {0, 1} untuk membentuk string 00101101, 010, …
  - Alfabet {a, b, c} untuk membentuk string-string ab, aabc, abcca, aaaaaaab,….
- Peringatan: Disini penulisan string tidak menggunakan tanda petik (apostroph) sehingga a dapat dibaca sebagai suatu simbol dan juga suatu string!

Variabel untuk String/Bahasa

- Dalam pembahasan suatu string (bahasa) bisa diwakili oleh sebuah variabel string (bahasa). Karena variabel memerlukan nama, maka kita perlu membuat konvensi penamaannya (jika akan menggunakan huruf-huruf huruf-huruf roman)
- **Simbol-simbol (**kalau bukan angka) menggunakan huruf-huruf kecil (font Arial) dari yang urutan terkiri: a, b, c, ...
- **Nama variable string** menggunakan huruf-huruf kecil italic: …, *x*, *y*, *z*. (huruf-huruf terkanan dalam abjad)
- **Variabel bilangan** menggunakan huruf-huruf: …, *k*, *l*, *m*, *n*, … (huruf-huruf menengah dalam abjad)
- **Bahasa** menggunakan nama dengan minimal huruf pertama huruf besar: L, A^nB^n, Pal, L_1, L_2.

Fungsi Property dari String: panjang, okurensi

- Jika *s* dan *t* merupakan string-string,
  - **Panjang string** *s* (notasi |*s*|): banyaknya simbol dalam string *s*.
    - |aaba| = 4 ; |ε| = 0
  - **Kemunculan/okurensi simbol** *c* dalam *s* (notasi $\#_c(s)$): jumlah kemunculan simbol *c* dalam *s*.
    - $\#_a(aaba) = 3$; $\#_a(bbc) = 0$

Operasi String: konkatenasi

- Jika *s* dan *t* merupakan string-string,
  - **Konkatenasi** *s* dan *t* (notasi *s*||*t*, atau lebih sederhana *st*): string yang didapat dengan menyambungkan *t* setelah *s*.
    - *s* = aab dan *t* = bb, maka *s*||*t* = aabbb ; |*st*| = |*s*| + |*t*|

Operasi String: replikasi, reversi

- Notasi w^n : **replikasi** string w sebanyak n kali, yang didefinisikan sbb:
  - $w^0 = \varepsilon$
  - $w^{n+1} = w^n w$

Contoh:
- a^3 = aaa
- (bye)2 = byebye
- a^0b^3 = bbb

- Notasi w^R : **reversi** dari w, yang didefinisikan
  - jika $|w| = 0$ maka $w^R = w = \varepsilon$
  - jika $|w| \geq 1$ maka
    - $\exists a \in \Sigma(\exists u \in \Sigma^* (w = ua))$, maka $w^R = au^R$
    - Contoh: $(\text{nama})^R$ = aman
  - Jika w dan x string, maka $(wx)^R = x^R w^R$
  - Contoh: $(\text{apakabar})^R = (\text{kabar})^R\ (\text{apa})^R$ = rabakapa

Relasi Antar String

- String *abaab* memiliki **prefiks-prefiks** ε, *a*, *ab*, *aba*, *abaa*, *abaab*
- String *abaab* memiliki **prefiks-prefiks sejati** (*proper prefix*) ε, *a*, *ab*, *aba*, *abaa*.
- String *abaab* memiliki **sufiks-sufiks** ε, *b*, *ab*, *aab*, *baab*, *abaab*
- String *abaab* memiliki **suffiks-sufiks sejati** (*proper suffix*) ε, *b*, *ab*, *aab*, *baab*
- String *abaab* memiliki **substring** *ba* tapi tidak memiliki substring *bab*

Lebih lanjut dengan Σ dan Σ^*

- Σ dapat merupakan alfabet dari beberapa bahasa, tetapi, bahasa-bahasa berbeda dapat memiliki alfabet berbeda (atau beririsan).
  - Jika itu terjadi kita gunakan subscript berbeda: Σ_1, Σ_2,....
- Σ^* adalah himpunan semua kemungkinan string yang dapat dibuat dari alfabel Σ,
  - setiap bahasa *L* dari Σ merupakan subset dari Σ^* (Σ^* adalah himpunan semesta semua *L* dari Σ).
  - Contoh: Jika $\Sigma = \{0, 1\}$, maka $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, \ldots\}$

Spesifikasi Bahasa

- Seperti halnya himpunan, jika L bahasa dengan alfabet $\Sigma = \{0, 1\}$, L dapat dispesifikasikan:
  - Dengan menyebutkan setiap anggotanya (Jika berhingga & berjumlah sedikit), misalnya
    $L = \{011, 1001, 11, 00110, 10011, 10, 110, 1101, 1110\}$
  - Dengan menderetkan beberapa anggota dan diikuti "…" agar anggota lain dapat diperkirakan, misalnya,
    $L = \{01, 0011, 000111, 00001111, \ldots\}$
  - Sebagai pasangan ekspresi dan predikat, terpisahkan dengan titik dua (":") atau garis tegak ("|"), misalnya.,
    - $L_1 = \{0^n1^n : n \geq 1\}$
    - $L_2 = \{w \in \Sigma^* : w \text{ memiliki prefiks } 000\}$
    - $L_3 = \{v111w : v, w \in \{0\}^*\}$

Contoh

- $L = \{x \in \{a, b\}^* :$ semua a mendahului b$\}$
- ε, a, aa, aabbb, dan bb adalah string pada L.
- aba, ba, and abc bukan string pada L.

Finite atau Countably-infinite

- Sebagai himpunan bahasa bisa *finite* atau *countably infinite*
  - ***Finite* (berhingga)**: jika string-string dalam bahasa bisa dienumerasi tuntas (hingga n dan $n < \infty$)
    Contoh: $L_1 = \{a^n b : n \leq 7\} = \{b, ab, \ldots, aaaaaaab\}$
  - ***Countably infinite***: jika terdapat bijeksi (pemetaan) dari setiap anggotanya ke bilangan natural (1,2,3…)
  - Contoh: $L_2 = \{a^n b : n \geq 0\} = \{b, ab, aab, aaab, \ldots\}$

Catatan: Dalam L_1 walaupun *n* besar sekali string terpanjang tetap bisa didapatkan, sementara dalam L_2 kita tidak bisa mendapatkan string terpanjangnya.

Uncountable?

- Suatu himpunan bisa:
  - Himpunan bilangan real antara 0 dan 1 adalah uncountable karena deretan bilangannya tidak dapat dipetakan ke bilangan bulat (disebutkan satu persatu).
  - Himpunan dari semua himpunan bagian dari suatu himpunan countably infinite adalah uncountable.

- Suatu bahasa dari alfabet {a,b} tidak uncountable tetapi countable infinite karena kita selalu bisa mengurutkan satu demi satu, misalnya secara proper-order (lihat slide berikutnya)!

Operasi Himpunan pada Bahasa

- Karena bahasa adalah himpunan string maka setiap operasi himpunan dapat diaplikasikan untuk bahasa-bahasa juga:
  - $L = L_1 \cup L_2$ **iff** $L = \{w : w \in L_1$ atau $w \in L_2\}$
  - $L = L_1 \cap L_2$ **iff** $L = \{w : w \in L_1$ dan $w \in L_2\}$
  - $L = L_1 - L_2$ **iff** $L = \{w : w \in L_1$ dan $w \notin L_2\}$
  - Himpunan semesta adalah Σ^*.
  - Jika L_1 merupakan komplemen L (dinotasikan $\neg L$), maka $L_1 = \{w \in \Sigma^* : w \notin L\}$
  - Komplemen dari Σ^* adalah $\varnothing$.
  - Dst…

Proper Order Bahasa

- **Proper order:** enumerasi string dengan cara
  - Urutan dari yang terpendek hingga terpanjang
  - $(\forall x(\forall y\ ((|x| < |y|) \rightarrow (x <_L y))$
  - String-string dengan panjang yang sama urutkan secara **lexicographic.**
- **Lexicographic order:** enumerasi string yang memenuhi $(c_1w_1 <_L c_2w_2)$ **iff** $(c_1 < c_2)$ atau $((c_1 = c_2)$ dan $(w_1 <_L w_2))$
  - Peringatan: dalam buku Rich, penulis mendefinisikan **Proper Order** sebagai **Lexicographic Order!**
- Contoh: L = {$w \in$ {0,1}* | w memiliki sufiks 100},

  proper order dari L adalah:

  100, 0100, 1100, 00100, 01100, 10100, 11100, ...

Cobalah sendiri!

- Dapatkanlah 5 anggota pertama secara proper order dari $L = \{w \in \{0,1\}^*$: w tidak memiliki substring 00$\}$
- Dapatkanlah 5 anggota pertama secara proper order dari $L = \{w \in \{0,1\}^*: \#_1(w)$ bilangan genap$\}$
- Dapatkanlah 5 anggota pertama secara proper order dari $L = \{w \in \{0,1\}^*$: setiap kemunculan 0 dalam w akan segera diikuti 11$\}$
- Dapatkanlah 5 anggota pertama secara proper order dari $A^nB^n = \{a^m b^n: 3m+2n > 5\}$

Kardinalitas Bahasa

- **Kardinalitas** Bahasa L adalah banyaknya **string berbeda** dalam L (notasi: $|L|$).
  - Bahasa dengan alfabet Σ yang memiliki kardinalitas terkecil adalah $\varnothing$ → kardinalitas 0.
  - Bahasa dengan alfabet Σ yang memiliki kardinalitas terbesar adalah Σ^* → kardinalitas ∞.
- Tanya: Berapa kardinalitas dari $L = \{x \in \Sigma^* : |x| \leq n\}$?
  - $|L| = |\{x \in \Sigma^* \mid |x| = 0\}| + |\{x \in \Sigma^* \mid |x| = 1\}| + \ldots |\{x \in \Sigma^* \mid |x| = n\}|$
  - $=1 + |\Sigma| + |\Sigma|^2 + \ldots + |\Sigma|^n$
  - Misalnya $\Sigma = \{0, 1\}$ dan $n = 5$, $|L| = 1 + 2 + \ldots + 32$

Konkatenasi Bahasa-bahasa

- Jika $L_1 \subseteq \Sigma_1^*$ dan $L_2 \subseteq \Sigma_2^*$, konkatenasi L_1 dengan L_2 atau $L_1L_2 = \{vw : v \in L_1 \text{ dan } w \in L_2\}$
  - Contoh:
    L_1 = {Kabupaten_, Kota_},
    L_2 = {Bandung, Bogor, Sukabumi},
    L_1L_2 = {Kabupaten_Bandung, Kabupaten_Bogor, Kabupaten_Sukabumi, Kota_Bandung, Kota_Bogor, Kota_Sukabumi}
- Sifat:
  - $L_1(L_2L_3) = (L_1L_2)L_3$
  - $|L_1L_2| = |L_1| \text{ x } |L_2|$
  - Jika L_1 atau L_2 countable infinite, maka L_1L_2 countable infinite.

Reversi Bahasa

- $L^R = \{w \in \Sigma^* : w = x^R \text{ untuk } x \in L\}$
- Sifat:
  - $(L_1 L_2)^R = L_2^R L_1^R$
  - $\varnothing^R = \varnothing$
- Catatan:
  - $L = \{w \in \Sigma^* : w = w^R\}$ dinamai juga bahasa Palindrom (Pal) dan memiliki sifat $L = L^R$.

Operasi-operasi Kleene* dan Kleene+

- Suatu bahasa dapat direplikasi (self-concatenated) pula,
  $L^n = LL...L$ (n kali), yang mana difenisikan bahwa:
  $L^0 = \{\varepsilon\}$
  $L^n = L^{n-1} L$
  $L^* = \{\varepsilon\} \cup$
  $\{w \in \Sigma^* : \exists k \geq 1$
  $(\exists w_1, w_2, \ldots w_k \in L \ (w = w_1 w_2 \ldots w_k))\}$

- Contoh:
  $L = \{$`dog`, `cat`, `fish`$\}$
  $L^* = \{\varepsilon,$ `dog`, `cat`, `fish`, `dogdog`,
  `dogcat`, `fishcatfish`,
  `fishdogdogfishcat`, ...$\}$

Operasi-Operasi Kleene* dan Kleene$^+$

- Operasi Kleene* (dibaca Kleene-star) didefinisikan sbb,
  - $L^* = U_{k \geq 0} L^k$
- Operasi Kleene$^+$ (dibaca Kleene-plus) didefinisikan sbb,
  - $L^+ = U_{k \geq 1} L^k$

- Tanya:
  - Benarkah $L^+ = LL^*$? (jawab: Ya, tapi mengapa?)
  - Karena $L^0 = \{\varepsilon\}$, benarkah selalu $L^+ = L^* - \{\varepsilon\}$? (jawab: Tidak! Coba cari kasus yang tidak memenuhi)

Contoh-contoh

- Apakah $\forall L.(L\varnothing = L)$? [Tidak, tapi mengapa?]
- Apakah $\forall L.(L\{\varepsilon\} = L)$? [Ya, tapi mengapa?]
- Apakah $\varnothing \cup \varnothing^* = \varnothing$? [Tidak, tapi mengapa?]
- Apakah $\forall L.((L^+)^* = L^*)$? [Ya, tapi mengapa?]
- Apakah $\{a, ab\}^* = (\{a\}^*\{ab\}^*)^*$? [Ya, tapi mengapa?]

Sifat Tertutup di bawah Suatu Operasi

- Bahasa L disebut "tertutup di bawah suatu operasi" (*closed under an operation*),
  - jika $\forall w \in L$, maka juga $F(w) \in L$.
- Contoh:
  - $L = \{w \in \{a, b\}^* : w$ memiliki sufiks $bba\}$, L tertutup di bawah operasi konkatenasi.
  - $L = \{w \in \{a, b\}^* : w$ memiliki sufiks $bba\}$, L tidak tertutup di bawah operasi reversi.
  - Pal tidak tertutup di bawah operasi konkatenasi.
  - Pal tertutup di bawah operasi reversi.

Ada Berapa Banyak Bahasa Berbeda dari Σ?

- Dengan alfabet Σ, bahasa berbeda yang bisa dibuat adalah sebanyak $|\mathcal{P}(\Sigma^*)|$ ($\mathcal{P}$ adalah power set).
  - Untuk $\Sigma = \emptyset$, karena $\Sigma^* = \varepsilon$, $\mathcal{P}(\Sigma^*) = \{\emptyset, \{\varepsilon\}\}$.
  - Untuk $\Sigma \neq \emptyset$, maka Σ^* countably infinite, berarti juga $\mathcal{P}(\Sigma^*)$ uncountably infinite.
    - (Mengapa? Lihat Theorem 1.2 dan 1.3, buku Elaine Rich).
- Note: berapa banyak bahasa berbeda dari Σ, dengan panjang string maksimum 10? Finite atau infinite?