MODUL 2: BAHASA REGULAR DAN EKSPRESI REGULAR

DEFINISI BAHASA REGULAR

Bahasa Regular L dari alfabet Σ adalah bahasa yang dapat dihasilkan dari bahasa-bahasa paling sederhana dari Σ dengan melakukan operasi-operasi gabungan, konkatenasi, dan/atau Kleene-*.

Apakah bahasa-bahasa paling sederhana dari Σ? Seperti yang dibahas dalam topik sebelumnya, bahasa-bahasa paling sederhana adalah bahasa-bahasa yang beranggota string tunggal dan string tersebut panjangnya satu atau nol.

Contoh: Bahasa dari {0,1} yang mana setiap anggotanya merupakan konkatenasi dari sejumlah substring 110 dan diakhiri oleh satu simbol apa saja (0 atau 1). Bahasa tersebut dapat dutuliskan sebagai ekspresi operasi himpunan $\{110\}^*\{0, 1\}$. Ternyata bahasa tersebut adalah regular karena ekspres operasi himpunan itu memenuhi definisi bahasa regular di atas sbb.

- konkatenasi {1}{1}{0} menjadi {110}
- Kleen {110} menjadi $\{110\}^*$
- penggabungan $\{0\} \cup \{1\}$ menjadi {0,1}
- konkatenasi hasil Kleen $\{110\}^*$ dengan hasil penggabungan {0,1} menjadi $\{110\}^*\{0,1\}$

Karena bahasa regular terbentuk dari bahasa-bahasa paling sederhana melalui ketiga jenis operasi tersebut maka setiap bahasa regular akan selalu dapat dispesifikasikan dengan suatu formula/notasi himpunan. Namun, notasi himpunan dalam beberapa hal masih kurang praktis. Berikut ini adalah notasi yang biasa digunakan untuk bahasa regular.

EKSPRESI REGULAR

Ekspresi regular bisa dibuat dari otasi himpunan dengan menghilangkan notasi kurung kurawal {} atau menggantikannya dengan kurung biasa, serta mengantikan notasi gabungan $\cup$ dengan notasi + menghasilkan ekspresi yang mirip dengan ekspresi aritmatis. Ekspresi ini disebut ekspresi regular.

Contoh:

Bahasa	**Ekspresi Regular dari Bahasa**
$\{\Lambda\}$	Λ
{0}	0
{001}	001
{0, 1}	0+1
{0, 10}	0+10
$\{1, \Lambda\}\{001\}$	$(1+\Lambda)001$
$\{110\}^*\{0,1\}$	$(110)^*(0+1)$
$\{1\}^*\{10\}$	1^*10
$\{10, 111, 11010\}^*$	$(10+111+11010)^*$
$\{0, 10\}^*(\{11\}^* \cup (001, \Lambda\})$	$(0+10)^*((11)^* + 001 + \Lambda)$

Bisa dikatakan bahwa ekspresi regular memperlihatkan pola string yang paling tipikal dari bahasa ybs. Contoh: 1^*10 merupakan string-string berisikan akhiran 10 dan diawali sejumlah 1.

Definisi Formal Ekspresi Regular

Kelas R dari bahasa regular pada Σ, dan ekspresi regularnya terdefinisi sebagai berikut:

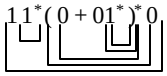
1. Bila $\varnothing$ merupakan anggota dari R, ekspresi regularnya adalah $\varnothing$.
2. Bila $\{\Lambda\}$ adalah anggota dari R, ekspresi regularnya adalah Λ.
3. Untuk setiap $a \in \Sigma$, $\{a\}$ adalah anggota dari R, ekspresi regularnya adalah a.
4. Bila L_1 dan L_2 adalah anggota R, sementara r_1 dan r_2 adalah ekspresi regularnya maka
   (a) $L_1 \cup L_2$ anggota R, dan ekspresi regularnya adalah $(r_1 + r_2)$
   (b) L_1L_2 anggota R, maka ekspresi regularnya adalah (r_1r_2)
   (c) L_1^* anggota R, maka ekspresi regularnya adalah $(r_1)^*$.
5. Hanya dengan keempat pernyataan di atas bahasa regular dapat diperoleh.

Adanya aturan pertama mengenai bahasa kosong $\varnothing$ diberikan untuk menjaga konsistensi dan kesederhanaan ekspresi. Sesuai dengan definisi himpunannya

maka untuk setiap ekspresi regular r_1 dan r_2 yang mana tepat salah satu $\varnothing$, misalnya $r_1 = \varnothing$, maka $r_1+r_2 = r_2$ dan $r_1r_2 = \varnothing$. Untuk lebih kompak lagi, kita bisa menggunakan ekspresi regular r^2 menggantikan rr, serta r^+ menggantikan r^*r.

Hirarki antar Operator

Seperti halnya dalam ekspresi aritmetis, untuk menghilangkan ambiguitas saat menggunakan notasi +, *, dan konkatenasi, maka diperlukan suatu hirarki penulisan notasi. Hirarki tertinggi adalah *, kemudian konkatenasi, dan terakhir +. Tanda kurung berperan untuk mengelompokkan suatu subekspresi sebagai satu entitas ekspresi seperti halnya penulisan formula aritmetis. Contoh:



Aljabar Ekspresi Regular

Keuntungan lainnya dari ekspresi regular ini adalah bahwa kita dapat menerapkan suatu aljabar ekspresi regular untuk penulisan ekspresi dalam bentuk yang lebih ringkas/sederhana. Contoh:

$1^*(1+\Lambda) = 1^*1 + 1^* = 1^+ + 1^* = 1^*$
$1^*1^* = 1^*$
$(0^*1^*)^* = (0+1)^*$
$(0+1)^*01(0+1)^* + 1^*0^* = (0+1)^*$

Persamaan pertama dan kedua sudah jelas. Persamaan ketiga dijelaskan sebagai berikut. 0^*1^* memiliki subset dengan ekspresi-ekspresi regular Λ, 0, 1 serta 0^+1^+, maka $0^*1^* = (\Lambda + 0 + 1 + 0^+1^+) = (0 + 1) + (\Lambda + 0^+1^+)$. Kemudian, $(0^*1^*)^* = ((0 + 1) + (\Lambda + 0^+1^+))^* = (0 + 1)^*(\Lambda + 0^+ + 1^+ + 0^+1^+)^*$. Karena $(0+1)^*$ merupakan himpunan semesta dari string-string yang terbentuk dari {0, 1} yang artinya sudah termasuk $(0 + 1)^*(\Lambda + 0^+1^+)^*$ maka terbukti $(0^*1^*)^* = (0+1)^*$.

Persamaan keempat dapat dibuktikan dengan analisis sebagai berikut. Suku $(0+1)^*01(0+1)^*$ menyatakan himpunan dari string yang selalu memiliki substring 01. String yang tidak memiliki substring 01 adalah string yang hanya berisi simbol 1 saja atau string yang hanya berisi simbol 0 saja atau jika keduanya ada maka simbol-simbol 0 akan berada setelah simbol-simbol 1. Ekspresi regular komplemen ini adalah 1^*0^*. Gabungan keduanya adalah himpunan seluruh string yang terbentuk dari {0,1} dengan ekspresi regular $(0+1)^*$.

Berikut ini sejumlah contoh penulisan bahasa regular dari deskripsi bahasa tsb.

Contoh 1: Bila $L \subseteq \{0,1\}^*$ merupakan bahasa dari semua string yang panjangnya genap ($L = \{x \in \{0,1\}^* \mid |x| \in$ bilangan genap}, dalam hal ini $\Lambda \in L$ karena 0 adalah genap).

Setiap string yang panjangnya genap adalah hasil konkatenasi dari string-string yang panjangnya dua. Jadi bahasa ini dapat ditulis: L = {00, 01, 10, 11}*. Ekspesi regular dari bahasa L adalah $(00+01+10+11)^* = (0(0+1)+1(0+1))^* = ((0+1)(0+1))^*$.

Contoh 2: Bila $L \subseteq \{0,1\}^*$ merupakan bahasa dari semua string yang berisi 1 dalam jumlah yang ganjil.

Setiap string dari L haruslah memiliki minimal satu 1. Jadi selalu terdapat berisikan satu 1 prefiks berbentuk 0^*10^* diikuti oleh nol atau lebih pasangan 1 sehingga ekspresi regularnya adalah: $0^*10^*(10^*10^*)^*$. Bisa juga dituliskan dengan $0^*1(0^*10^*1)^*0^*$ atau $(0^*10^*1)^*0^*10^*$ atau $0^*(10^*10^*)^*1(0^*10^*1)^*0^*$.

Contoh 3: Bila $L \subseteq \{0,1\}^*$ merupakan bahasa dari semua string yang panjangnya 6 atau kurang ($L = \{x \in \{0,1\}^* \mid |x| \leq 6\}$).

L = {Λ, 0, 1, 00, 01, 10, 11, 000, …, 111110, 111111} atau ekspresi regularnya adalah
$\Lambda+0+1+00+01+10+11+000+ \ldots +111110+111111 = (\Lambda+0+1)^6$

Contoh 4: Bila $L \subseteq \{0,1\}^*$ merupakan bahasa dari semua string yang diakhiri oleh 1 dan tidak memiliki substring 00.
Pernyataan bahwa tidak ada substring 00 serta 0 tidak ada diakhir string berimplikasi setelah setiap kemunculan 0 akan diikuti 1. Jadi ekspresi regular L adalah $(1+01)^+$. Kleene-$^+$ digunakan disini karena Λ tidak termasuk L.

Contoh 5: Bila *l* adalah ekspresi regular (`a+b+...+z+A+B+...+Z`) dan d adalah ekspresi regular (`0+1+...+9`) dan _ adalah karakter 'underscore'. maka setiap variabel dalam bahasa pemrograman C (setiap string dari huruf, angka dan underscore tapi diawali oleh huruf atau underscore) memikili eksrepsi regular:

$(l+\_)(l+d+\_)^*$

Contoh 6: Jika *p* adalah karakter titik dan *s* adalah ekspresi regular ($\Lambda + a + m$) dengan *a*=karakter plus, *m*=karakter minus, maka bilangan real dalam bahasa pemrograman Pascal memiliki ekspresi regular:

$sd^+ (pd^+ + pd^+Esd^+ + Esd^+)$